Automated muscle path calibration with gradient-specified optimization based on moment arm

Ziyu Chen, Tingli Hu, Sami Haddadin, *Fellow, IEEE*, David W. Franklin

***Abstract*— *Objective:* Muscle path modeling is more than just routing a cable that visually represents the muscle, but rather it defines how moment arms vary with different joint configurations. The muscle moment arm is the factor that translates muscle force into joint moment, and this property has an impact on the accuracy of musculoskeletal simulations. However, it is not easy to calibrate muscle paths based on a desired moment arm, because each path is configured by various parameters while the relations between moment arm and both the parameters and joint configuration are complicated. *Methods:* We tackle this challenge in the simple fashion of optimization, but with an emphasis on the gradient; when specified in its analytical form, optimization speed and accuracy are improved. *Results:* We explain in detail how to differentiate the enormous cost function and how our optimization is configured, then we demonstrate the performance of this method by fast and accurate replication of muscle paths from a state-of-the-art shoulder–arm model. *Conclusion and Significance:* As long as the muscle is represented as a cable wrapping around obstacles, our method overcomes difficulties in path calibration, both for developing generic models and for customizing subject-specific models.**

***Index Terms*— Chain rule, gradient, model calibration, moment arm, musculoskeletal model, muscle path, optimization**

I. Introduction

In musculoskeletal modeling, the geometry of a muscle is often represented by a series of straight and curved cables, namely the muscle path. It is defined by the locations of the origin, via, and insertion points as well as the size(s), location(s), and orientation(s) of the obstacle(s) around which the cables wrap [1]–[4]. Although such a path is a geometrical simplification of the muscle, which alters properties related to the 3-D architecture, it can still have similar muscle length–joint angle and moment arm–joint angle relations to its anatomical reference, provided the path is well configured [5]–[8]. These two relations are crucial to simulation accuracy since muscle length is a major variable in the contraction dynamics [9]–[13] and moment arm directly determines the kinetic capacity of a muscle [14]–[16].

The calibration of a muscle path is not only a fundamental step for the development of a generic musculoskeletal model, but also necessary in subject-specific applications [17]–[21]. Scaling of skeletal geometry, for example, is a common practice of model individualization, where the sizes and shapes of the subject's segments may be replicated in the model based on anthropometric measurements or inverse kinematics [2], [22]. However, such a scaled model is not yet subject-specific, because the individual characteristics of muscle length and moment arm are only reflected in musculoskeletal geometry, which is different from skeletal geometry [2], [12]. In fact, the scaling of skeletal geometry might distort the biomechanical characteristics which were calibrated to be generically correct. For example, depending on how the path of the triceps surae is defined, enlarging the tibia might result in the Achilles tendon moment arm being increased, unchanged, or even decreased. Thus, for a subject-specific model, a rework of path calibration is necessary after scaling to reflect the individual characteristics of muscle length and moment arm, or at least assure that they remain similar to those in the generic model.

Muscle path calibration is performed based on experimental data such as geometric coordinates from medical images [17]–[19], [23] or moment arm measurements [6], [24], and this process can be laborious. To begin with, there is a large number of path-related parameters to be tuned. Each muscle path is defined by at least six parameters, including the two 3-D coordinates of the origin and insertion points. In addition, there will be three extra parameters for each via point and many more when obstacles are included to recreate the anatomical feature of muscle geometry [5], [18]. For example, in the prevalent open-source modeling platform OpenSim [22], cylinders are often used as obstacles to prevent the muscles from penetrating into the joints [7], [24], [25], where the cylinder size, orientation, and location would respectively need one, two, and three parameters to define. In general, monoarticular muscles require between six to 12 parameters, whereas complicated multiarticular muscles may require up to 30 parameters [18].

The size of parameters is not necessarily a big challenge when one knows which to tune. However, if an objective

This work was supported by the Lighthouse Initiative Geriatronics by StMWi Bayern (Project X, grant no. 5140951) and the German Federal Ministry of Education and Research funding (Project AI.D, grant no. 16ME0539K).

All authors are with the Munich Institute of Robotics and Machine Intelligence (MIRMI), Technical University of Munich (TUM), Germany.

Tingli Hu and Sami Haddadin are also with the Chair of Robotics and Systems Intelligence (RSI), TUM School of Computation, Information and Technology, Germany

Ziyu Chen and David W. Franklin are also with Neuromuscular Diagnostics, TUM School of Medicine and Health, Germany. David W. Franklin is also with the Munich Data Science Institute (MDSI), TUM, Germany. (e-mail: david.franklin@tum.de)

function is too complex, the sensitivity to each parameter is often unclear [21]. Hence, manual tuning tends to be puzzling: e.g., enlarging the obstacle or shifting it away from the center of rotation might increase the corresponding moment arm, but the effect may differ when the joint configuration changes. Even with optimization, the tuning process is not labor-free, if weighting factors are involved and need to be tested repeatedly due to the absence of any knowledge regarding the sensitivity.

The second challenge in muscle path calibration lies in the high dimensional relation between moment arm and joint configuration. Each muscle, even monoarticular ones, can actuate multiple degrees of freedom (DoF), and each DoF corresponds to one moment arm [26]–[28]. For instance, the gastrocnemius is a biarticular muscle crossing the knee and ankle, which can be considered to contain at least three DoFs (ankle plantar-/dorsiflexion, ankle eversion/inversion, knee extension/flexion). Therefore, besides the well-studied Achilles tendon moment arm in the sagittal plane [29]–[31], it has an extra ankle moment arm [32], [33] and a moment arm about the knee [26], [34]. With all DoFs considered, path calibration involves matching multiple moment arm–joint angle relations altogether, and it is common to run into trouble where the calibration of one moment arm can distort others that are previously calibrated.

There is more to this challenge than only the high dimensional relation, because each moment arm is also affected by all actuating DoFs of the muscle [32], [35], [36]. Suppose the knee moves while the ankle is immobilized, in theory the Achilles tendon ankle moment arm might still change despite no ankle motion. With this, the moment arm–joint angle relation is no longer depictable by a curve or a surface, but rather requires a hypersurface to demonstrate. For example, the soleus has two moment arms around the ankle, and trying to calibrate them is similar to matching two pairs of surfaces (imagine two heatmaps), which is difficult but viable. However, for the gastrocnemius, each of its three moment arms is dependent on three DoFs, which means that depicting the relation of each moment arm with the DoFs is similar to plotting a volumetric heatmap in 3-D space. In this case, the task of calibration becomes matching the color for three pairs of 3-D heatmaps. While difficult to imagine, manual calibration is theoretically still possible if they are somehow matched slice by slice, where each slice is a 2-D heatmap. But with one more moment arm, e.g. for the rectus femoris or many of the shoulder muscles, the number of dimensions to be matched is beyond three, forcing manual tuning to be simplified and compromising the overall model accuracy.

In light of these challenges, it takes extensive effort to develop musculoskeletal models to simulate many different motions or to individualize them for each subject. This difficulty hinders the advancement and application of musculoskeletal modeling and simulation. Therefore, here we develop a gradient-based method for automated muscle path calibration. Our goal is to tune path-related parameters so that the moment arm–joint angle relation of a model matches with the target specification. An optimization framework is established for a classic muscle path wrapping method [1] and the gradient for the cost function is derived in its analytical form to increase optimization speed and accuracy. The concept employed in the gradient derivation is universal and may be applied to the model calibration of many other complex systems.

II. Methods

A. Optimization Method

The process of parameter tuning for a muscle path is formulated as a least-squares problem with the cost function

$$J(\boldsymbol{p}) = \sum_{n=1}^{N} \left\| \boldsymbol{r}_{\text{target}}(\boldsymbol{q}_n) - \boldsymbol{r}_{\text{model}}(\boldsymbol{q}_n, \boldsymbol{p}) \right\|_2^{\,2} \to \min, \quad (1)$$

where $\boldsymbol{p}$ denotes muscle path parameters, and $\boldsymbol{r}(\boldsymbol{q})$ is the moment arm at joint configuration $\boldsymbol{q}$. Note that $\boldsymbol{r}$ and $\boldsymbol{q}$ are vectors with the dimension of DoF number, whereas N is the number of joint configurations. The target value and the model output are indicated by the subscripts, but to make the notations concise, the model output will also be abbreviated as $\boldsymbol{r}(\boldsymbol{q}, \boldsymbol{p})$, $\boldsymbol{r}(\bullet, \boldsymbol{p})$, or $\boldsymbol{r}$.

In this study, we limited the obstacle type to cylinder to simplify the discussion, but the same principle applies to other types such as sphere and sphere-capped cylinder described in [1]. Currently, our optimization method requires the composition of each muscle path to be decided in advance, which is based on three fundamental segment types: straight (no obstacle), single-cylinder, and double-cylinder. A path can be constructed as an individual segment or a combination of multiple different segments, e.g.,

$$\underbrace{\text{origin—}\overbrace{\text{via—cylinder—via}}^{\boldsymbol{p}_{\text{single}}}}_{\boldsymbol{p}_{\text{straight}}}\underbrace{\text{—cylinder—cylinder—insertion}}_{\boldsymbol{p}_{\text{double}}},$$

and the composition of $\boldsymbol{p}$ is correspondingly based on three fundamental forms:

1) $\boldsymbol{p}_{\text{straight}}$: $({}^{\text{j}}\boldsymbol{u}_{\text{P}}, {}^{\text{j}}\boldsymbol{u}_{\text{S}})$
2) $\boldsymbol{p}_{\text{single}}$: $({}^{\text{j}}\boldsymbol{u}_{\text{P}}, {}^{\text{j}}\boldsymbol{u}_{\text{S}}, R, {}^{\text{j}}\boldsymbol{u}_{\text{C}}, \alpha, \beta)$
3) $\boldsymbol{p}_{\text{double}}$: $({}^{\text{j}}\boldsymbol{u}_{\text{P}}, {}^{\text{j}}\boldsymbol{u}_{\text{S}}, R_1, {}^{\text{j}}\boldsymbol{u}_{\text{C}_1}, \alpha_1, \beta_1, R_2, {}^{\text{j}}\boldsymbol{u}_{\text{C}_2}, \alpha_2, \beta_2)$

where ${}^{\text{j}}\boldsymbol{u}_{\text{P}}, {}^{\text{j}}\boldsymbol{u}_{\text{S}} \in \mathbb{R}^3$ are the coordinates of the anchor points of a path segment expressed in their reference frames, written as column vectors; e.g., ${}^{\text{j}}\boldsymbol{u}_{\text{P}} = [{}^{\text{j}}x_{\text{P}} \;\; {}^{\text{j}}y_{\text{P}} \;\; {}^{\text{j}}z_{\text{P}}]^{\text{T}}$. The radius of a cylinder is denoted by $R \in \mathbb{R}^+$ and the center by ${}^{\text{j}}\boldsymbol{u}_{\text{C}} \in \mathbb{R}^3$, whereas the orientation is defined by two Euler angles $\alpha, \beta \in \mathbb{R}$. In the following discussion, the start and end anchor points of a path segment are denoted as P and S respectively, and when wrapped by the cylinder (whose center is denoted as C), the two wrapping points are denoted as Q and T (Fig. 1). Also, the frame in which coordinates are expressed are indicated by the superscript: with w for the world frame, c for the cylinder frame, and j for one of the joint frames; see Appendix A for coordinate transformation.

A muscle path is configured by $\boldsymbol{p}$ using the obstacle-set method [1], which computes the potential wrapping points on the obstacle(s) by finding the minimum-distance path between the two anchor points in each segment. To reduce computational load, we first took a geometric approach to compute $\boldsymbol{r}$, which is based on the principle of virtual work and is algorithmically efficient by computing the velocity terms

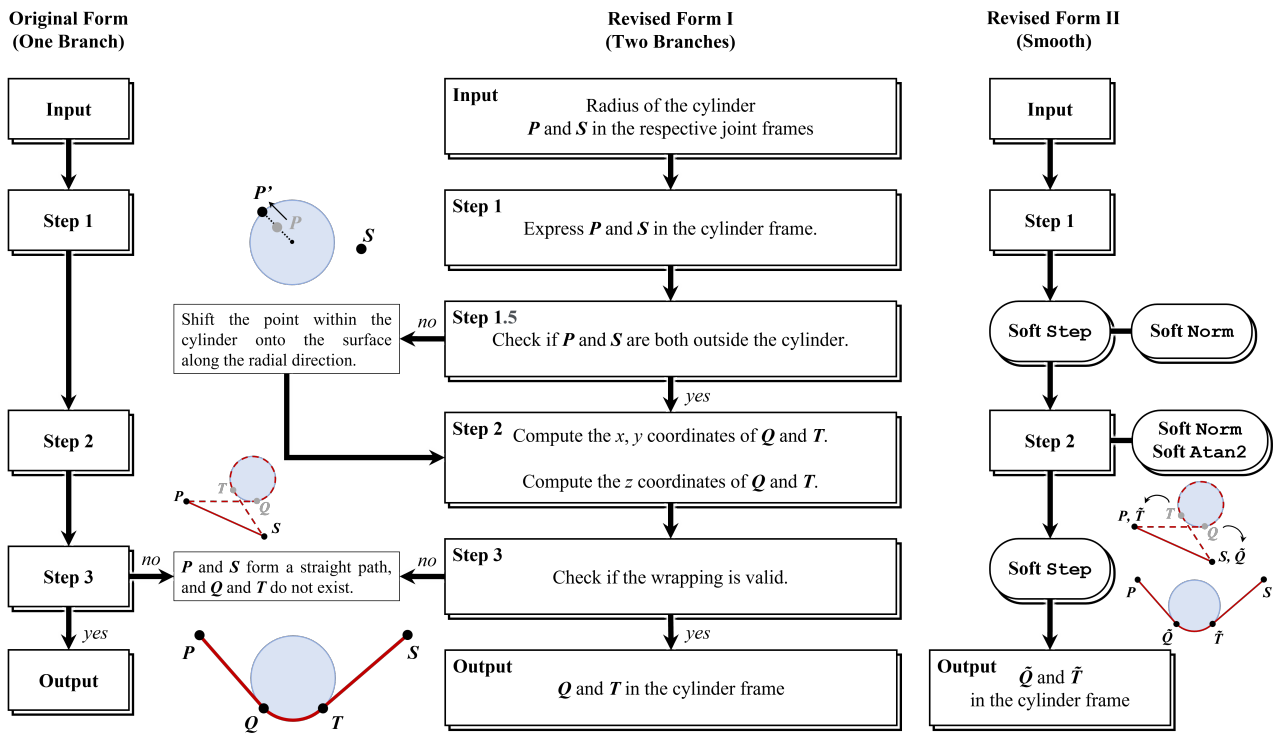


Fig. 1. Workflows of the original and revised obstacle-set methods. Left: the original form with one conditional statement. Middle: the revised form with an extra conditional statement for continuity. Right: the smooth form revised for optimization. The naming and order of the anchor and wrapping points in the single-cylinder case are indicated by the illustration in the bottom.

using the Kane's method [37]–[39]. Then, since the typical algorithms solving (1) (e.g., Levenberg-Marquardt or trust-region) require the gradient $\partial J/\partial \boldsymbol{p}$, and by the chain rule

$$\frac{\partial J}{\partial \boldsymbol{p}} = \frac{\partial J}{\partial \boldsymbol{r}}\frac{\partial \boldsymbol{r}}{\partial \boldsymbol{p}} = \sum_{n=1}^{N}\left(2\Big(\boldsymbol{r}(\boldsymbol{q}_n,\boldsymbol{p}) - \boldsymbol{r}_{\text{target}}(\boldsymbol{q}_n)\Big)\right)\frac{\partial \boldsymbol{r}}{\partial \boldsymbol{p}}, \quad (2)$$

we need to specify $\partial \boldsymbol{r}/\partial \boldsymbol{q}$ in its analytical form.

Notice for a path composed of I segments, its length (l) is the sum of the lengths of all segments, and the same applies to its moment arm ($\partial l/\partial \boldsymbol{q}$). This means (2) can be decomposed as individual computations for each path segment with

$$\begin{aligned} \boldsymbol{r}(\bullet,\boldsymbol{p}) &= \sum_{i=1}^{I}\boldsymbol{r}_i(\bullet,\boldsymbol{p}_i), \\ \frac{\partial \boldsymbol{r}}{\partial \boldsymbol{p}} &= \sum_{i=1}^{I}\frac{\partial \boldsymbol{r}_i}{\partial \boldsymbol{p}} = \sum_{i=1}^{I}\Big(\frac{\partial \boldsymbol{r}_i}{\partial \boldsymbol{p}_i}\frac{\partial \boldsymbol{p}_i}{\partial \boldsymbol{p}}\Big), \end{aligned} \quad (3)$$

where the i-th segment is configured by $\boldsymbol{p}_i$, either in the form of $\boldsymbol{p}_{\text{straight}}$, $\boldsymbol{p}_{\text{single}}$, or $\boldsymbol{p}_{\text{double}}$. Therefore, essentially our goal is to specify $\partial \boldsymbol{r}_i/\partial \boldsymbol{p}_i$, and knowing that the case of any multi-segment path can be broken down to an individual segment, our following discussion of moment arm focuses on the path segment and we drop the subscript i for convenience.

Direct derivation of $\partial \boldsymbol{r}/\partial \boldsymbol{p}$ is overwhelming considering its structural complexity. It would require a tremendous amount of effort, and the eventual result would be filled with pages of repeated terms and computationally inefficient. Thus, we circumvented this problem by disassembling it into a composite of multiple gradients whose analytical forms are easy to derive. For instance, moment arm can be computed given the anchor and wrapping points (as in $\boldsymbol{r}(\boldsymbol{u})$), and by the chain rule

$$\frac{\partial \boldsymbol{r}}{\partial \boldsymbol{p}} = \frac{\partial \boldsymbol{r}}{\partial \boldsymbol{u}}\frac{\partial \boldsymbol{u}}{\partial \boldsymbol{p}}, \quad (4)$$

where $\boldsymbol{u}(\boldsymbol{p})$ is obtained with the obstacle-set method. As we will show later, $\boldsymbol{r}(\boldsymbol{u})$ is relatively simple in structure, so $\partial \boldsymbol{r}/\partial \boldsymbol{u}$ is not difficult to derive, and our remaining goal is to compute $\partial \boldsymbol{u}/\partial \boldsymbol{p}$. However, $\boldsymbol{u}(\boldsymbol{p})$ is still complex and contains indifferentiable components, so we began with revising the obstacle-set method into a smooth form, such that $\partial \boldsymbol{u}/\partial \boldsymbol{p}$ exists for all $\boldsymbol{p}$. This is accomplished by replacing the conditional statements in the obstacle-set method as well as other nondifferentiable components with soft functions which yield almost the same outputs but are continuously differentiable (Fig. 1).

B. Soft Functions

Fig. 1 (middle) shows the detailed workflow of the revised obstacle-set method, containing two conditional statements. The first statement checks if either of the anchor points is inside the cylinder. Note that this was not included in the original method by [1] (Fig. 1, left), and we introduced it to simplify the optimization since it removes complex geometric constraints between the anchor points and the obstacle. The second statement checks if the wrapping angle exceeds 180°, which is considered invalid wrapping. Naturally, conditional statements make $\boldsymbol{r}(\bullet,\boldsymbol{p})$ nondifferentiable in certain domains, and we introduced soft functions to patch them (Fig. 1, right).

Step 1.5 essentially keeps a minimum distance of R between the point and the cylindrical axis (e.g., $\|{}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{P},xy}\|_2 = \sqrt{{}^{\mathrm{c}}x_{\mathrm{P}}{}^2 + {}^{\mathrm{c}}y_{\mathrm{P}}{}^2}$) to prevent Step 2 from returning `Inf` or `NaN`. In our case, if for example P locates inside the cylinder, it is shifted onto the surface along the radial direction as P′ (Fig. 1, middle), and the coordinates become

$$ {}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{P}',\mathrm{xy}} = \frac{R}{\|{}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{P},xy}\|_2}\,{}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{P},xy}. \tag{5} $$

More specifically,

$$ {}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{P}},\mathrm{xy}} = \begin{cases} {}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{P},xy}, & \|{}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{P},xy}\|_2 > R \\ {}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{P}',\mathrm{xy}}, & \|{}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{P},xy}\|_2 \le R \end{cases}, \tag{6} $$

and the coordinates of such transitional point $\tilde{\mathrm{P}}$ may be calculated using a soft version of Heaviside step function:

$$ {}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{P}},\mathrm{xy}} = {}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{P}',\mathrm{xy}} + \left({}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{P},\mathrm{xy}} - {}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{P}',\mathrm{xy}}\right) f_{\mathrm{softStep}}(d_{\mathrm{P}}), \tag{7} $$

where $d_{\mathrm{P}} = {}^{\mathrm{c}}x_{\mathrm{P}}{}^2 + {}^{\mathrm{c}}y_{\mathrm{P}}{}^2 - R^2$, and

$$ f_{\mathrm{softStep}}(x) = \frac{1}{e^{-2x}+1} \approx \begin{cases} 1, & x > 0 \\ 0, & x < 0 \end{cases}; \tag{8} $$

the input can be magnified to steepen the transition around 0. Now, the conditional statement of (6) is smoothed: $\tilde{\mathrm{P}}$ remain as P if $d_{\mathrm{P}} > 0$, otherwise it approaches the shifted point P′ when d_{P} tends to 0 in the negative direction.

With ${}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{P}}}$ and ${}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{S}}}$, Step 2 finds the appropriate points of tangency Q and T on the cylinder (${}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{Q}}$ and ${}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{T}}$; see (S11) and (S13) in Appendix C). Then based on the sign of

$$ s = {}^{\mathrm{c}}x_{\mathrm{Q}}{}^{\mathrm{c}}y_{\mathrm{T}} - {}^{\mathrm{c}}x_{\mathrm{T}}{}^{\mathrm{c}}y_{\mathrm{Q}}, \tag{9} $$

Step 3 returns either the coordinates of the two wrapping points or null if wrapping is invalid (Fig. 1, left). The latter situation can be problematic as it leads to yet another conditional statement in moment arm computation, in which different formulas are used depending on whether the wrapping occurs:

$$ \text{straight: } \boldsymbol{r} = \left({}^{\mathrm{jP}}\boldsymbol{\omega}_{\tilde{\mathrm{P}}} - {}^{\mathrm{jS}}\boldsymbol{\omega}_{\tilde{\mathrm{S}}}\right)^{\mathrm{T}} \frac{{}^{\mathrm{w}}\boldsymbol{u}_{\tilde{\mathrm{S}}} - {}^{\mathrm{w}}\boldsymbol{u}_{\tilde{\mathrm{P}}}}{\|{}^{\mathrm{w}}\boldsymbol{u}_{\tilde{\mathrm{S}}} - {}^{\mathrm{w}}\boldsymbol{u}_{\tilde{\mathrm{P}}}\|_2}, \tag{10} $$

$$ \begin{aligned} \text{wrapped: } \boldsymbol{r} = &\left({}^{\mathrm{jP}}\boldsymbol{\omega}_{\tilde{\mathrm{P}}} - {}^{\mathrm{jC}}\boldsymbol{\omega}_{\mathrm{Q}}\right)^{\mathrm{T}} \frac{{}^{\mathrm{w}}\boldsymbol{u}_{\mathrm{Q}} - {}^{\mathrm{w}}\boldsymbol{u}_{\tilde{\mathrm{P}}}}{\|{}^{\mathrm{w}}\boldsymbol{u}_{\mathrm{Q}} - {}^{\mathrm{w}}\boldsymbol{u}_{\tilde{\mathrm{P}}}\|_2} \\ &+\left({}^{\mathrm{jC}}\boldsymbol{\omega}_{\mathrm{T}} - {}^{\mathrm{jS}}\boldsymbol{\omega}_{\tilde{\mathrm{S}}}\right)^{\mathrm{T}} \frac{{}^{\mathrm{w}}\boldsymbol{u}_{\tilde{\mathrm{S}}} - {}^{\mathrm{w}}\boldsymbol{u}_{\mathrm{T}}}{\|{}^{\mathrm{w}}\boldsymbol{u}_{\tilde{\mathrm{S}}} - {}^{\mathrm{w}}\boldsymbol{u}_{\mathrm{T}}\|_2}, \end{aligned} \tag{11} $$

where $\boldsymbol{\omega}$ is obtained using (S2) in Appendix A.

Here, the revision of Step 3 into a smooth form involves a transition from (10) to (11) when s changes from positive to negative, and the complication arises with the involvement of joint frames: P, S, and C could be fixed on separate bones; i.e., the implication of $\mathrm{j_P}$, $\mathrm{j_S}$, and $\mathrm{j_C}$ may be different. This makes it hard to cancel out the terms in (11) to transition into (10). To this end, we have Q and T respectively approaching S and P when s tends to 0 in the negative direction (Fig. 1, right), and similarly this can be achieved with (8):

$$ \begin{aligned} {}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{Q}}} &= {}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{S}}} + \left({}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{Q}} - {}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{S}}}\right) f_{\mathrm{softStep}}(s), \\ {}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{T}}} &= {}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{P}}} + \left({}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{T}} - {}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{P}}}\right) f_{\mathrm{softStep}}(s). \end{aligned} \tag{12} $$

This way, moment arm computation is generalized as

$$ \begin{aligned} \boldsymbol{r} = &\left({}^{\mathrm{jP}}\boldsymbol{\omega}_{\tilde{\mathrm{P}}} - {}^{\mathrm{jC}}\boldsymbol{\omega}_{\tilde{\mathrm{Q}}}\right)^{\mathrm{T}} \frac{{}^{\mathrm{w}}\boldsymbol{u}_{\tilde{\mathrm{Q}}} - {}^{\mathrm{w}}\boldsymbol{u}_{\tilde{\mathrm{P}}}}{\|{}^{\mathrm{w}}\boldsymbol{u}_{\tilde{\mathrm{Q}}} - {}^{\mathrm{w}}\boldsymbol{u}_{\tilde{\mathrm{P}}}\|_2} \\ &+\left({}^{\mathrm{jC}}\boldsymbol{\omega}_{\tilde{\mathrm{T}}} - {}^{\mathrm{jS}}\boldsymbol{\omega}_{\tilde{\mathrm{S}}}\right)^{\mathrm{T}} \frac{{}^{\mathrm{w}}\boldsymbol{u}_{\tilde{\mathrm{S}}} - {}^{\mathrm{w}}\boldsymbol{u}_{\tilde{\mathrm{T}}}}{\|{}^{\mathrm{w}}\boldsymbol{u}_{\tilde{\mathrm{S}}} - {}^{\mathrm{w}}\boldsymbol{u}_{\tilde{\mathrm{T}}}\|_2}. \end{aligned} \tag{13} $$

When $s > 0$, $\tilde{\mathrm{Q}}$ and $\tilde{\mathrm{T}}$ maintain their original coordinates as wrapping points Q and T (${}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{Q}}} \approx {}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{Q}}$ and ${}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{T}}} \approx {}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{T}}$), and moment arm is computed with (11). When $s < 0$, ${}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{Q}}} \approx {}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{S}}}$ and ${}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{T}}} \approx {}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{P}}}$, and (13) simplifies into (10). Importantly, this generalization accommodates the wrapping and moment arm computation involving two obstacles.

Furthermore, Step 2 involves the calculation of the 2-norm (also in (5) and (13)) and the 2-argument arctangent (see (S13) in Appendix C). Both are nondifferentiable at the origin, and we replaced them with

$$ f_{\mathrm{softNorm}}(\boldsymbol{x}) = \sqrt{\|\boldsymbol{x}\|_2{}^2 + a} - \sqrt{a}, \tag{14} $$

and

$$ \begin{aligned} &f_{\mathrm{softAtan2}}(y,x) = 2\arctan\left(\frac{\gamma}{2}\right) \\ &\frac{\gamma}{2} = \begin{cases} \frac{-b}{\sqrt{x^2+b^2}+x}, & -b < y < 0 \text{ and } x < 0 \\ \frac{b}{\sqrt{x^2+b^2}+x}, & 0 \le y < b \text{ and } x < 0 \\ \frac{y}{\sqrt{x^2+y^2}+x}, & \text{otherwise} \end{cases}, \end{aligned} \tag{15} $$

where a and b should be sufficiently small positive numbers (e.g., `eps` and 0.001 respectively).

In such a manner, we have modified a branched workflow into a smooth form that is differentiable across the entire domain of its input parameters. This modification is verified using kinematic measurements to ensure that its difference with the non-smooth form is small enough across a large range of motion (see Appendix B).

With (8), the obstacle-set method (i.e., the computation of $\boldsymbol{u}(\boldsymbol{p})$) is revised into a smooth form, and since (8), (14), and (15) are clearly all differentiable, it is now possible to compute the gradient $\partial\boldsymbol{u}/\partial\boldsymbol{p}$. As shown in (4), our target function $\boldsymbol{r}(\bullet,\boldsymbol{p})$ is a composite of $\boldsymbol{u}(\boldsymbol{p})$ and the generalized moment arm computation $\boldsymbol{r}(\bullet,\boldsymbol{u})$ with (13). However at this stage, $\partial\boldsymbol{r}/\partial\boldsymbol{p}$ is still difficult to compute due to the knotty structure of $\partial\boldsymbol{u}/\partial\boldsymbol{p}$, so we continue down the chain; see Appendix C for the mathematical derivation in detail.

C. Calibration and Validation

With the gradient $\partial\boldsymbol{r}/\partial\boldsymbol{p}$ specified, we used the nonlinear least-squares solver (`lsqnonlin`) in MATLAB to solve (1). The computation of this cost function requires a musculoskeletal model and the input of some kinematic dataset $\boldsymbol{q}$ for the model output, and the correspondent moment arm data at $\boldsymbol{q}$ are required as the target value. Ideally, $\boldsymbol{q}$ should cover joint configurations as diverse as possible for accurate calibration.

For the model output, we used a 12-DoF 42-muscle human shoulder–arm model [40] (with muscle path from [18]; see all DoFs and muscle paths in Appendix D), and the kinematic measurements of the shoulder and arm from 10 intransitive

daily tasks performed by a single subject in [41]. The kinematics from five tasks (gesturing an OK sign, pumping fists, blocking out light from the face, saluting, and pointing) were input to the reference model to generate artificial calibration data, while $\boldsymbol{q}$ from another five tasks (gesturing a thumb-down, signaling for hitchhike, greeting, gesturing to stop, and gesturing for silence) were used for validation.

For the target value, we utilized the same model as reference to generate artificial moment arm data; that is, (1) becomes

$$J(\boldsymbol{p}) = \sum_{n=1}^{N} \left\| \boldsymbol{r}_{\text{modelGP}}(\boldsymbol{q}_n, \boldsymbol{p}_{\text{ref}}) - \boldsymbol{r}_{\text{modelGP+}}(\boldsymbol{q}_n, \boldsymbol{p}) \right\|_2^2 \to \min, \tag{16}$$

where $\boldsymbol{r}_{\text{modelGP}}$ is computed based on the slightly revised obstacle-set method by [1] (Fig. 1, middle) with $\boldsymbol{p}_{\text{ref}}$ from [18], while $\boldsymbol{r}_{\text{modelGP+}}$ is based on our smoothed form (Fig. 1, right). Thus in a sense, this process is equivalent to replicating the muscle path geometry in the reference model. The input of $\boldsymbol{q}$ was interpolated from the aforementioned kinematic measurements. This is because the original dataset consists of joint configurations from a massive number of time instants (1000-Hz sampling rate), many configurations are similar and hence redundant as input for producing the target value. For calibration, to avoid an underdetermined system, the number of joint configurations equals to the number of path parameters that each muscle has (Supplementary Table II); e.g., an 18-parameter path will be calibrated based on the kinematics in a total of 18 instants from five movements (N as 18 in (16)). For validation, five instants were extracted from each movement, that is 25 joint configurations in total.

With the model output and target value in place, we set out to demonstrate the performance of our method through a comparison test as well as the implementation of muscle path calibration. In the test, the solver was configured to run with and without the gradient specified analytically, and we compared the optimization results for some representative muscle paths. In order to take algorithmic features into account, the comparison between gradients was repeated for two `lsqnonlin` algorithms. In the implementation, the solver was tasked to calibrate the moment arms of all 42 muscle paths, and we manipulated the input using four variants representing progressive levels of application complexity. The input variants were generated based on a 2×2 design: original/modified parameter structure × noise-free/noisy calibration data. For both the test and the implementation, optimization was performed on a 2.9-GHz Intel Core i9 with 64 GB RAM and 14 CPU cores using parallel computing (`parfor`): The processor was not overclocked, and the number of parallel workers was set as 14. We structured parallel computing to optimize 14 initial points simultaneously for a single muscle path within each `parfor` loop. Other details are described as follows.

The comparison test aims to evaluate the contribution of gradient specification as well as to determine an appropriate configuration for the implementation. First, we tested the `levenberg-marquardt` algorithm on two single cylinder–based (serratus anterior, superior and middle parts) and two double cylinder–based muscle paths (latissimus dorsi, thoracic and iliac parts), each with 70 sets of initial points. The `levenberg-marquardt` algorithm is relatively simple in that it has only one type of search direction [42], [43], and the optimization process should generally be similar when the analytical and numerical gradients differ only by numerical error. So with this test, we may also verify if our computation of the analytical gradient is correct. The initial points for each path—including locations (mm), radius (mm) and Euler angles (rad)—were randomly generated using `rand` in five ranges (from $[-1, 1]$ to $[-10^4, 10^4]$ with an increment of magnitude in between) and optimized without any constraints for our method to be verified across a vast domain. Then, we tested `trust-region-reflective`, which is the default algorithm for `lsqnonlin` and more robust. Optimization was performed with two additional complex paths (extensors carpi radialis brevis and ulnaris), each with 84 sets of initial points. They were also randomly generated but were bounded with slight anatomical constraints—the same initialization that will be explained later for the implementation. For both tests, `FunctionTolerance`, `OptimalityTolerance`, and `StepTolerance` were configured as 2.5×10^{-16} with `MaxIterations` as 10^3 and `MaxFunctionEvaluations` as 10^5, and the numerical gradient was computed using MATLAB's built-in forward-difference method.

Based on the preliminary results, the implementation of calibrating 42 muscle paths was configured as follows:

- `trust-region-reflective` (gradient-specified);
- 10^{-4} for `StepTolerance`, and 2.5×10^{-16} for `FunctionTolerance` and `OptimalityTolerance`;
- 10^2 for `MaxIterations` and `Inf` for `MaxFunctionEvaluations`,

and the initial points were generated with the subsequent considerations.

As previously mentioned, each muscle path requires predetermination of the number of via points and cylinders for the structure of $\boldsymbol{p}$. Some muscle paths are modeled with via points and more than two obstacles (Supplementary Table II), and their structures of $\boldsymbol{p}$ will be a combination of the three typical forms. For example, the extensor carpi radialis longus is modeled in the form of

$$\underbrace{\text{origin—via}}_{\boldsymbol{p}_{\text{straight}}}\overbrace{\text{—cylinder—via}}^{\boldsymbol{p}_{\text{single}}}\underbrace{\text{—via}}_{\boldsymbol{p}_{\text{straight}}}\overbrace{\text{—cylinder—via}}^{\boldsymbol{p}_{\text{single}}}\underbrace{\text{—insertion}}_{\boldsymbol{p}_{\text{straight}}},$$

which contains a total of 30 path parameters. To test the robustness of our method against parameter structure, we considered two scenarios. First, we set the structure of path parameters to the same as in the reference model. This is mostly the case of subject-specific modeling, where the parameter structure is known in the generic model and calibration is performed to adjust parameter values. In the second scenario, all muscle paths were configured with either one or two cylinders and without any via points, resulting in a different parameter structure from their reference counterpart (Supplementary Table II) and approximately 10% fewer parameters in total ($\boldsymbol{p}$ and $\boldsymbol{p}_{\text{ref}}$ in (16) may differ in dimension). In other words, with at most two obstacles to wrap around, this path needs to imitate

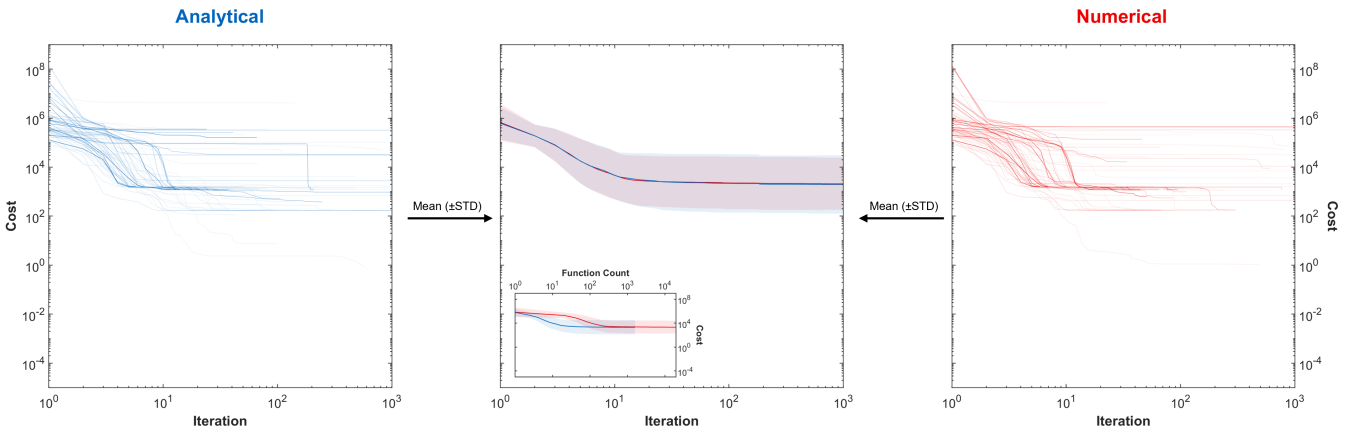


Fig. 2. Cost–iteration curves for the optimization using the `levenberg-marquardt` algorithm (latissimus dorsi, iliac part). The results of each initial point from the analytical (blue) and numerical (red) gradients are plotted respectively on the left and right. The mean and STD of the costs are plotted in the middle, with function count–dependent curves in the corner.

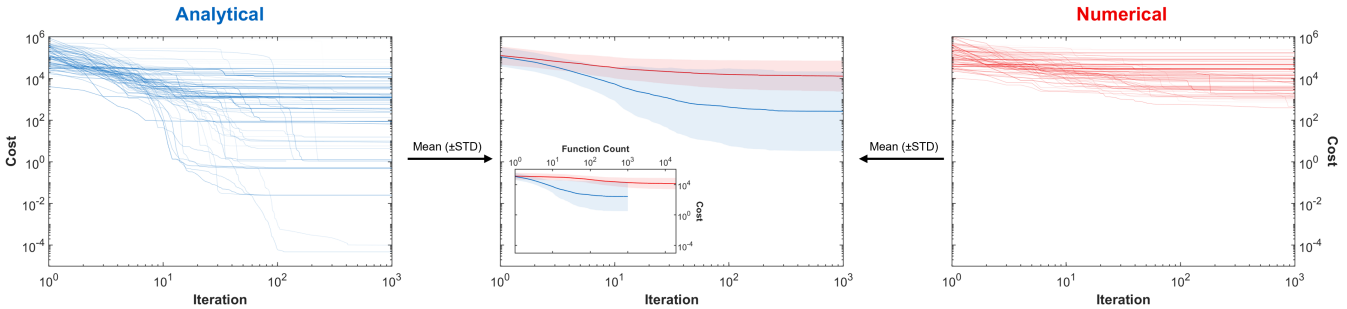


Fig. 3. Cost–iteration curves for the optimization using the `trust-region-reflective` algorithm (latissimus dorsi, iliac part). The results of each initial point from the analytical (blue) and numerical (red) gradients are plotted respectively on the left and right. The mean and STD of the costs are plotted in the middle, with function count–dependent curves in the corner.

the geometry of the original path, potentially configured by multiple points and obstacles. This is similar to the case of generic modeling, where often one obstacle per joint is placed regardless of how complex the muscle geometry might be.

Apart from the predetermination of parameter structure, We bounded the joint-frame coordinates of the origin and insertion points (not all anchor points) within a uniformly distributed range of ±30 mm from the respective original values in the reference model—in both initialization and optimization. This is not a prerequisite, but since the origin and insertion points attach to anatomical landmarks with accurate measurements, we included this constraint to speed up the calibration. Also, when generating the initial points, the joint-frame coordinates of cylinder center(s) were randomly initialized around the sections of the line between the initial origin and insertion points. This also accelerates the process, since if a cylinder is not wrapped at the beginning of the optimization, $\partial \boldsymbol{r} / \partial {}^{\mathrm{j}}\boldsymbol{u}_{\mathrm{C}}$ remains a null matrix, and ${}^{\mathrm{j}}\boldsymbol{u}_{\mathrm{C}}$ might be left untuned till the end. For efficiency, we ensured that the cylinders are not too departed from the initial muscle paths.

To reduce the risk of local minimum, the optimization was globally iterated with multiple randomly generated initial points. For each path, the number of initial points is 14 times the number of parameters (Supplementary Table II); e.g., a 12-parameter path will be calibrated with at most 168 sets of different initial values. Additionally, to realize a trade-off between speed and accuracy, we programmed the calibration to first iterate over 42 initial points (i.e., three `parfor` loops), and if the cost (16) per moment arm per joint configuration does not reach a sufficiently small level k (e.g., 0.01), another 42 will be iterated. The rest of initial points will only be iterated when the normalized cost fails to reach $100k$ after the first 84 global iterations, and the calibration stops whenever $100k$ is reached or when all initial points run out. If none of the global iterations reaches the desired threshold, 14 sets of initial points that led to the lowest costs but had the solver exit due to the iteration limit will be further optimized with `MaxIterations` as 10^3.

Optimization performance is also influenced by the calibration data, thus to examine the robustness of our method against error, the calibration was also performed with noisy data. For this, the aforementioned artificial moment arm data were added to with a composite of relative error (uniformly distributed within ±20% of the reference value) and absolute error (uniformly distributed within ±2 mm). This leads to a total of four conditions for the implementation (original/modified parameter structure × noise-free/noisy calibration data), and each was simulated five times with different sets of initial points to evaluate the performance in terms of calibration speed and validation accuracy. Note that the five sets of initial

points were kept the same for both noise-free and noisy conditions, and stopping criteria remained unchanged for all conditions; otherwise, the difference in results might also be attributed to initialization and configuration.

The calibration results are validated by mean absolute error

$$\boldsymbol{\epsilon}_{\text{val}} = \frac{\sum_{n=1}^{25} \left| \boldsymbol{r}_{\text{modelGP}}(\boldsymbol{q}_n, \boldsymbol{p}_{\text{ref}}) - \boldsymbol{r}_{\text{modelGP}}(\boldsymbol{q}_n, \boldsymbol{p}_{\text{opt}}) \right|}{25}, \quad (17)$$

where $\boldsymbol{p}_{\text{opt}}$ is the solution of (16). Here, notice the optimized parameters are input into the non-smooth model $\boldsymbol{r}_{\text{modelGP}}$ despite obtainment from the smooth form, making the evaluation standard more strict.

III. Results

We selected a few representative muscle paths to test the optimization with two classic `lsqnonlin` algorithms as well as to compare the performance with and without gradient specification. Using the `levenberg-marquardt` algorithm, the optimization processes based on the analytical and numerical gradients are generally the same, with costs descending in identical fashions for most of the initial points (Fig. 2 and Supplementary Figs. 3–5). Whereas with `trust-region-reflective`, the cost descents much faster with gradient-specified optimization (Fig. 3 and Supplementary Figs. 6–10): On average, by the 100th or even the 10th iteration, the cost minimized based on the analytical gradient is already lower than the final cost achieved by the numerical gradient.

We also recorded the computation time and the number of function counts (how many times the cost function is evaluated) for each iteration. Fig. 4 shows the ratio of computation time per iteration between the gradient-specified and unspecified optimizations, and as can be expected based on the computational principle of the numerical gradient, this ratio is almost linear to the number of parameters in the cost function. With this ratio, we scaled the cost descent to the number of function counts in Figs. 2 and 3, and Supplementary Figs. 3–10 to demonstrate optimization performance with respect to computation load. From this perspective, with `levenberg-marquardt`, even if the cost reduction per iteration is almost the same between the gradient-specified and unspecified optimizations, the former requires fewer function evaluations and is hence more efficient. As for the `trust-region-reflective` algorithm, the advantage achieved by the analytical gradient is further magnified by this computational efficiency.

To demonstrate the performance of our method in the implementation, we devised four conditions in which 42 muscle paths are calibrated of their multi-dimensional moment arms, and the results are summarized in Table I and visualized in Fig 5 and Supplementary Figs. 11–13. In an ideal condition where the parameters-to-optimize share the same structure with their reference and the calibration data contain no noise, our method completed the calibration task in 7.7 min with parallel computing. Across all 182 effective moment arms, the median validation error is 5×10^{-5} mm and the max is 3.37 mm for the palmarflexion/dorsiflexion moment arm of the flexor carpi radialis.

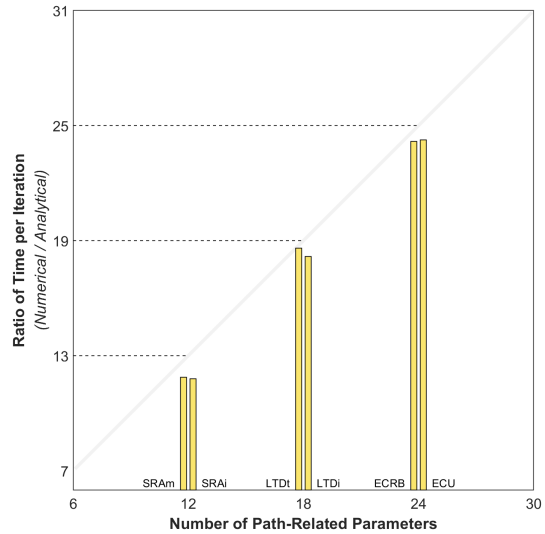


Fig. 4. Ratio of computation time per iteration (numerical vs. analytical) of six representative muscle paths. See Supplementary Table II for abbreviations.

When the input parameters are modified of their structures (e.g., a path defined by two cylinders is now calibrated with only one cylinder, or a path with multiple via points is now calibrated without them), the calibration time is 10.9 min, and only one moment arm contains error more than 5 mm in validation. When the calibration data are noisy, the performance of our method is reduced, which is not surprising since the introduced noise is quite large (a composite of ±20% relative error and ±2 mm absolute error). Nevertheless, its speed and accuracy are still satisfactory. For example, in the case of the original parameter structure, the calibration task was completed in 56.9 min with only one moment arm containing validation error over 10 mm.

For a more intuitive understanding of our method's working process, we demonstrate in Figs. 6 and 7 two cases of muscle path calibration, each with a featured complication. In Fig. 6, the initial anchor points are both within the cylinder. This is a situation that the original obstacle-set method cannot handle, and we patched the issue by shifting such a point onto to the cylinder surface to prevent a breakdown in wrapping point computation. As soon as the optimization starts, they are gradually expelled out of the cylinder to reduce the cost, achieving a smooth transition from one branch to another within a conditional statement. Fig. 7 shows a typical path configured by two cylinders, and the misplacement of either cylinder will induce error in moment arm computation. Driven by the gradient, first the orientations of both cylinders are rotated to the correct directions, and then their radii are adjusted to the right sizes—all in tens of iterations.

IV. Discussion

A key to optimization speed and accuracy is the gradient, which in our case is the gradient of the moment arm function with path-related parameters as variables. To derive it analytically, the cost function must be differentiable in the first place. For this, we revised the obstacle-set method for muscle

TABLE I
METHOD PERFORMANCE.

Condition		Calibration Speed		Validation Accuracy				
Param. Struct.	Calib. Data	Computing Time (min)		Mean Absolute Error [ϵ_{val}] (mm)			Moment Arm Count[1]	
		Serial	*Parallel*	*Median*[1]	*95-pct*[1]	*Max*[1]	ϵ_{val}>5 mm	>10 mm
Original	Noise-free	68.9±3.7	7.7±0.3	5e-5	1.20	3.37	0	0
Modified	Noise-free	92.4±6.3	10.9±0.9	0.09	2.62	5.58	1	0
Original	Noisy	468.6±8.1	56.9±0.4	1.87	6.74	10.17	19	1
Modified	Noisy	340.0±6.9	45.3±0.5	1.79	6.09	10.50	17	2

[1]Calculated or counted from the 182 effective moment arms of all 42 muscle paths.

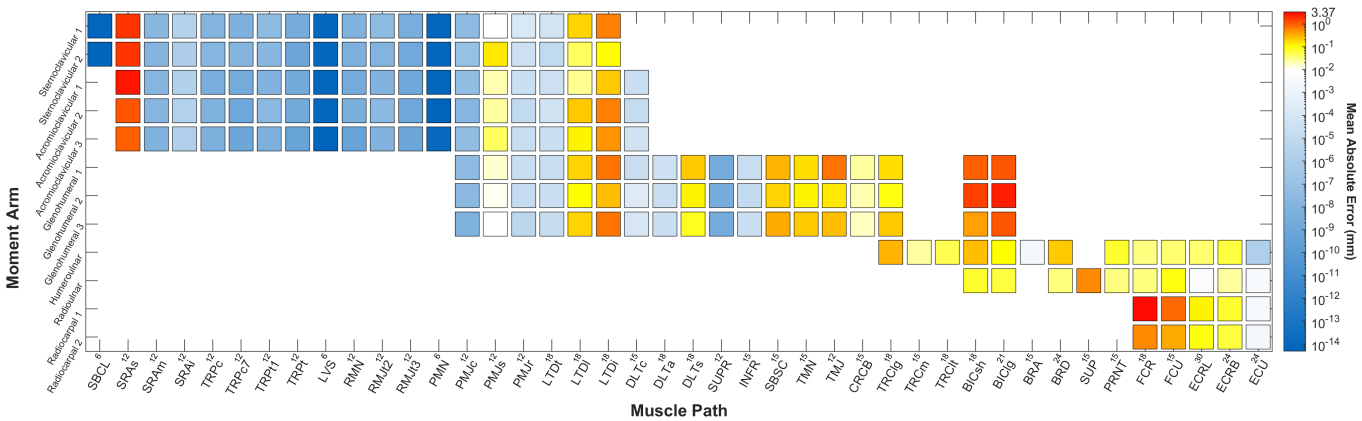


Fig. 5. Absolute errors in 182 effective moment arms of 42 muscle paths in validation (original parameter structure and noise-free calibration data). The magnitude of the error is indicated by the intensity of the color. The number in the horizontal axis corresponds to the parameter number for each muscle path. See Appendix D for nomenclatures in the axes.

path wrapping by substituting its conditional statements and nondifferentiable components with soft functions. Next, we disassembled the revised function into smaller modules that are easier to derive separately, and then assembled the gradients back into the desired gradient composite. The result is simplistic in form and thus easy to compute, which further increases optimization speed. Importantly, the concept of the chain rule is universal and may be useful for deriving the gradient of other complex functions as well.

Specifying the gradient in its analytical form is more than a formal alternative to gradient approximation using the typical finite difference method. As we demonstrate in our preliminary tests, the performance of the gradient-specified optimization is distinct from that of the gradient-unspecified. This arises from how the analytical and numerical gradients are computed as well as how each optimization algorithm proceeds with different gradients. For example, to estimate the gradient numerically with the forward-difference method, consider

$$\frac{\partial J(\boldsymbol{p})}{\partial \boldsymbol{p}} \approx \left[\frac{J(\boldsymbol{p}+\delta \mathbf{e}_1)-J(\boldsymbol{p})}{\delta} \quad \cdots \quad \frac{J(\boldsymbol{p}+\delta \mathbf{e}_n)-J(\boldsymbol{p})}{\delta} \right], \tag{18}$$

where $\mathbf{e}_i$ is the standard basis vector with the i-th element as 1 and 0s elsewhere, and δ is a sufficiently small positive value. To approximate the gradient, the additional number of function evaluations equals to the dimension of input parameters; this number will double if a central-difference method is used. In other words, the time complexity of (18) is $O\big(\dim(\boldsymbol{p})\big)$ [44]–[46]. However, if the analytical form is explicit, the cost function only needs to be evaluated once, since most terms for computing the gradient analytically are already accessible as part of the cost function evaluation; time complexity being $O(1)$. As shown in Fig. 4, the average computation time per iteration based on the numerical gradient is approximately $(\dim(\boldsymbol{p})+1)$ times of that based on the analytical, so the optimization is always much faster if the gradient is specified.

As for how the optimization progresses each iteration, the specifics depend on the algorithm. For instance, the `levenberg-marquardt` algorithm has only one conditional statement, in which a search direction–related factor is either magnified or reduced based on if a step (i.e., an iteration) succeeds in reducing the cost [43]. Consequently, the residual between the analytical and numerical gradients will not induce any difference in the optimization, unless the accumulated numerical error happens to diverge a step from success to failure. For most of the initial points we tested, the optimization process is identical between the analytical and numerical gradients (Fig. 2 and Supplementary Figs. 3–5). This also verifies our derivation of the analytical gradient.

With the `trust-region-reflective` algorithm, there is a much larger difference in the results between the gradient-specified and unspecified optimizations (Fig. 3 and Supplementary Figs. 6–10). The reason is that this algorithm not only contains a conditional statement with three branches that could separate the optimization process, but these branches are unique in terms of determining the direction of a search step: When the Hessian of the cost function is positive definite, the Newton's method is applicable for rapid local convergence

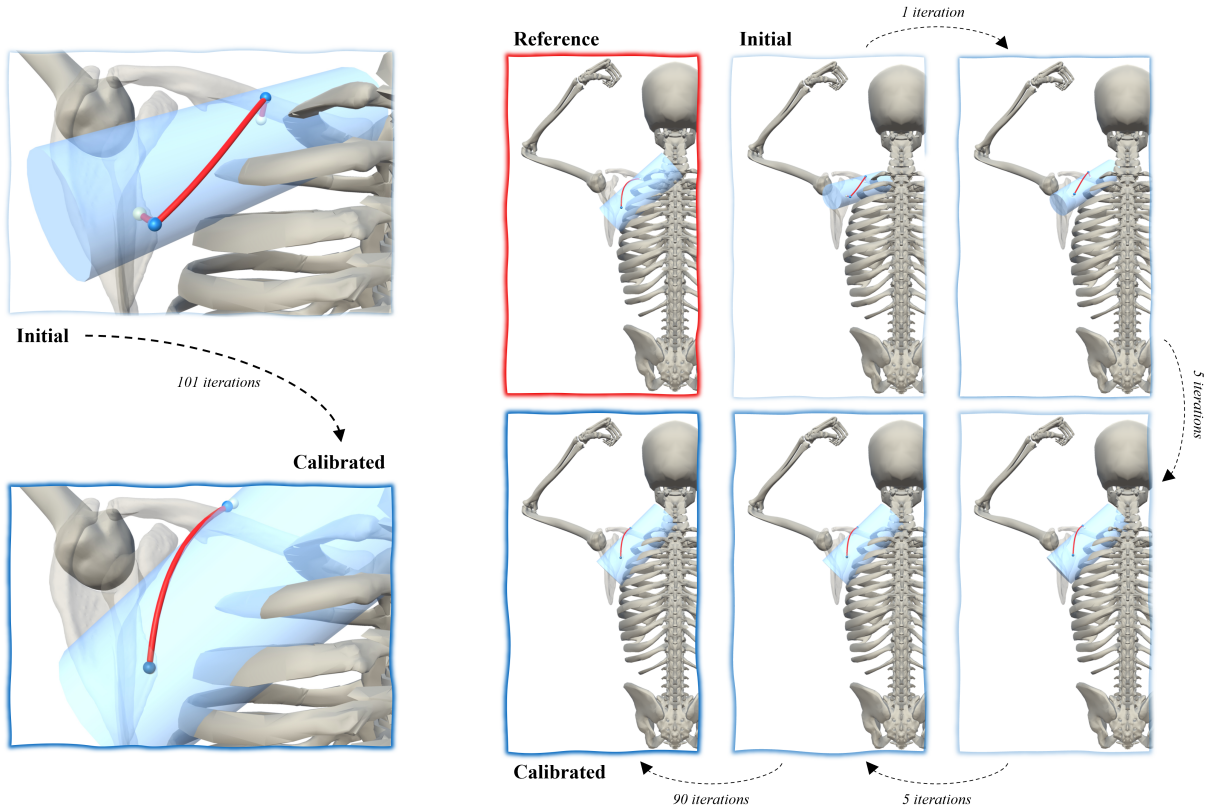


Fig. 6. A featured case of path calibration for the serratus anterior (superior part). Left: the anchor points and the obstacle defined by pre- and post-optimized parameters and the resultant wrapping points. Right: the reference path configuration and the optimization process from the initial point to the solution. The yellow points on the red muscle path are the anchor points, while the wrapping points are in blue. The initial anchor points are inside the cylinder, which would induce a computation breakdown in the original obstacle-set algorithm but is patched by our revision. As the optimization progresses, the anchor points are gradually expelled from the cylinder, leading to a path configuration valid for the original algorithm.

(namely a step in the Newton direction), otherwise the solution must be approached either in the direction of the steepest descent or negative curvature [42], [43]. Since the Hessian is approximated using quasi-Newton methods [42], if the gradient itself is already a numerical estimation, the Hessian approximation might not be positive definitive even if its exact value is. In this case, a Newton step is no longer possible, and neither of the alternatives converges to the solution as quickly. With the analytical gradient, however, the approximation is closer to the exact Hessian. Based on our observation so far, for every iteration, the Hessian approximated based on the analytical gradient maintains positive definite, indicating that Newton steps are likely taken throughout the optimization even with alternatives in `trust-region-reflective`, hence rapid convergence is ensured. Notably, the positive definiteness of the Hessian also allows the application of conjugate gradient methods to obtain each Newton step [42], avoiding the need for inverting the Hessian using Gaussian elimination, which is computationally less efficient.

To encapsulate, specifying the gradient in its analytical form benefits the optimization in two ways:

1) efficient computation of every iteration, and
2) rapid convergence (i.e., fewer iterations) in trust-region methods.

These two advantages grant the gradient-specified optimization much more cost descent per function count compared with the unspecified, allowing the search from many more initial points for the global solution. For instance, this rapidly convergent iteration process can be utilized for selection and mutation in genetic algorithm [46]–[48] and guarantees producing the locally optimal child without generating a large population.

With these advantages, the specification of gradient resolves most difficulties for the optimization in practice, but not all of them. Ideally, had the solver run for enough time with a sufficiently large number of iterations, there is not much need for any further conditions or constraints. However, expenditures in computation and time are always a concern in reality, and we may as well include some conditions as long as the required information is not more than we already have.

To begin with, the type of obstacle as well as the number are predetermined in our method to define the vector structure of path parameters. It is of course possible to introduce a soft function that merges moment arm computation with different amount of various obstacles, so that the obstacle type and number can also be optimized. Yet we find it unnecessary at the current stage, since the cylinder is a more popular obstacle for muscle path modeling, and the obstacle number is highly related to the number of joints a muscle crosses so it can be easily predetermined.

We also constrained the 3-D coordinates of the origin and insertion points to a range (±30 mm) around the reference

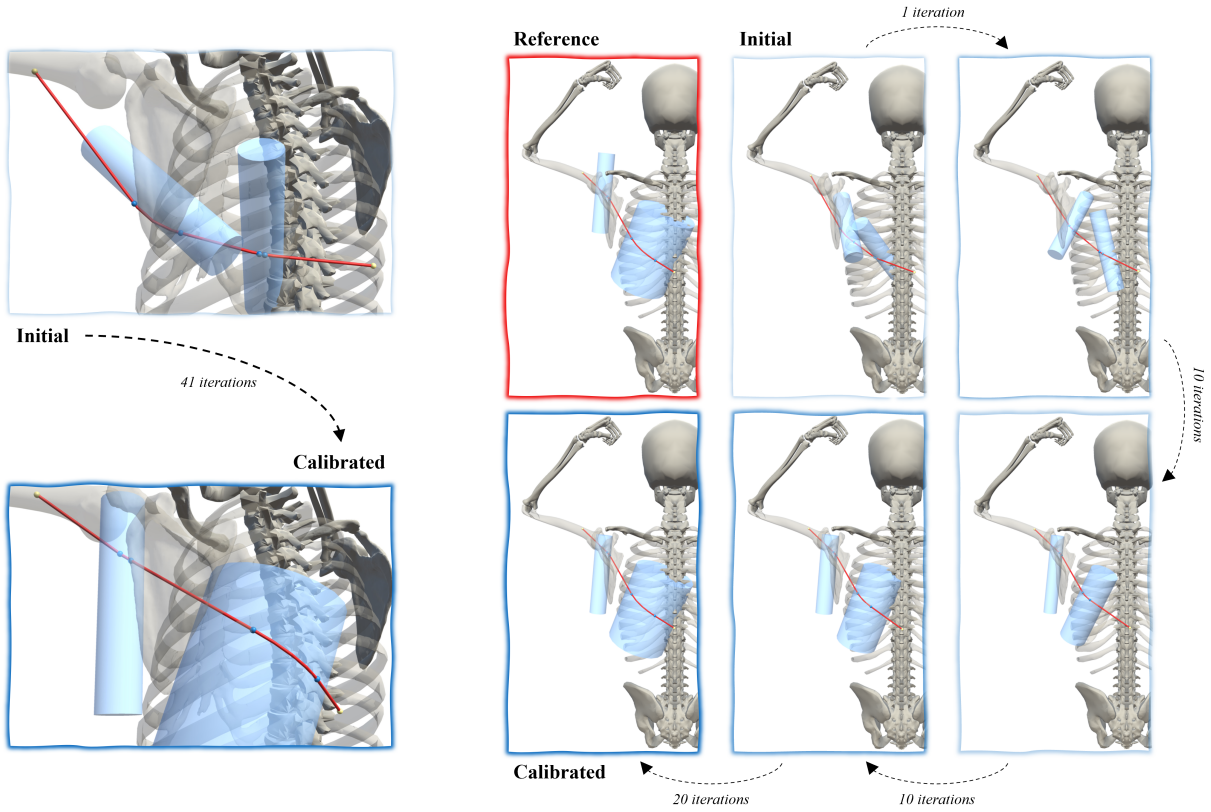


Fig. 7. A featured case of path calibration for the latissimus dorsi (thoracic part). Left: the anchor points and the obstacles defined by pre- and post-optimized parameters and the resultant wrapping points. Right: the reference path configuration and the optimization process from the initial point to the solution. The yellow points on the red muscle path are the anchor points, while the wrapping points are in blue. The path is complicated in that it wraps around two obstacles, both of which need to be correctly placed for moment arm calculation to be accurate. By the 11th iteration, the cylinders are oriented to the correct directions, and their radii are adjusted correctly in another 30 iterations. The parameters related to cylinder placement are optimized without constraints.

values; imagine the volume of a standard Rubik's Cube. This is not essential for our method, but it is also not necessary to spend time optimizing what is already quite clear. The origin and insertion are attached to anatomical landmarks, whose relative locations on the bones are well-studied. Especially if the goal is to calibrate upper-limb muscle path, even with individual variability taken into account, the origin and insertion points should not shift too much from our selected bounds. Other parameters, particularly those of the cylinders, were tuned without any constraint, so the major complication of obstacle placement is still being tackled without the knowledge of musculoskeletal geometry (Fig. 7). In fact, our modification of the obstacle-set method has already removed complex geometric constraints between the cylinder and the points, and the optimization may proceed from any initial point (Fig. 6).

Last, there is the typical problem of local minimum, which cannot be avoided even with the optimization gradient and constraints. For this, we adopted the common strategy of multiple initial points, and for each muscle, 84–420 different initial points could be iterated in optimization. This does not eliminate the risk of local minimum, but should avoid it as much as possible. Again, it would be neat and tempting to hit the global minimum with only one shot, but our approach is more practical with the computational cost taken into account.

The performance of our method is demonstrated by replicating the muscle paths of a reference model in four conditions (Table I). Here, note that we validated the optimized parameters using non-smooth wrapping and moment arm computation, meaning that these parameters are (sub-)optimal for the original model as well. It is hence practical to implement our method for other models: For example, with some slight modifications to accommodate other wrapping strategies (not necessary if the outcome is similar), it is possible to build upon our method a user-friendly program, which takes in any desired skeleton model and moment arm–joint angle relations for muscle path calibration reproducible in OpenSim and other software.

Moreover, we need to reiterate that it is mainly the high dimensional relation between moment arm and joint configuration that makes muscle path calibration challenging, which is often neglected. The moment arm in our reference model is not a one-dimensional variable but a vector of 12 elements. For instance, the pectoralis major crosses the shoulder complex and actuates up to eight DoFs, and each of the accordant moment arms varies with changes in any one of the eight DoFs, meaning that their moment arm–joint angle relation is depicted in a 8-D space. Calibrating the moment arms of the pectoralis major is similar to matching eight pairs of 8-D hypersurfaces, and it is almost impossible to accomplish manually unless four or five moment arms are neglected. In

spite of that, our method succeeds in calibrating most muscles with satisfactory speed and accuracy: e.g., the three parts of the pectoralis major were each calibrated in a few seconds, and the validation errors in their 24 moment arms are all negligible.

One limitation of this study is that the performance of our method is demonstrated only with artificial data generated from a model. To eliminate the concern that our optimization performance mainly benefits from the same structure shared by the parameters in the reference model and those to be calibrated, we also tested the scenario where a simpler parameter structure is configured. As shown in Table I and Supplementary Fig. 11, though both calibration speed and validation accuracy decrease compared with when the original parameter structure is configured, the calibration time is only 10.9 min and the median validation error is 0.09 mm. In reality, the path of a muscle is geometrically far more complicated than a few points and wrapping obstacles, but our results show that the optimization could still succeed with a relatively simple parameter structure. We also tested conditions with noisy data, and our method is quite robust (Table I and Supplementary Figs. 12 and 13) against errors likely larger than expected in experimental measurements. Importantly, our method is conveniently compatible with experimental data, since the only mandatory input is the relationship between moment arms and joint angles, regardless of how the data are obtained or whether the subject is human.

Another expected concern could be that, due to current technical limits, we usually do not have access to the measurements for all moment arms of a muscle, let alone their relations with all actuating DoFs. Yet this is not a problem related to our method but rather faced by all biomechanists. When lacking measurements, the moment arms will simply be assumed of certain values or relations, which is essentially what has been done in musculoskeletal modeling. Nevertheless, had such advanced measurements been available, our method is in line to provide equally advanced calibration. In fact, even if the high dimensional moment arm–joint angle relation is not completely established, our method is already usefully applicable. For example, to calibrate the Achilles tendon plantarflexion moment arm for a subject-specific model, one may use ultrasound or MRI to perform the measurement at the neutral ankle position, and then linearly scale some generic moment arm relation [49], [50] to obtain $\boldsymbol{r}_{\text{target}}(\boldsymbol{q})$ for our method. Or if one assumes some relation between moment arm and subject anthropometry, then they can also take moment arm values in the generic model as reference, scale them based on the assumed relation, and input in our method to generate a subject-specific model. In either case, the resultant muscle path will be much more accurate than when path parameters are directly scaled based on skeletal geometry.

Given these limitations, we expect our method to be examined with more models and datasets. As aforementioned, a practical application of this method would be in subject-specific modeling to recalibrate muscle path after scaling, which will be demonstrated in our future work. We are currently utilizing this method to develop a lower-limb model based on experimental data, which should offer further validation aligned with the theoretics in the current study.

It is also worth noticing that, apart from the key role in muscle path calibration, our analytically derived gradient reveals conclusively the sensitivity of moment arm to path-related parameters. The value of each element in the gradient denotes how much influence each parameter has on moment arm in a certain joint configuration, which could be of important reference in medicine. For example, in the surgery repairing rotator cuff tears, the positional error of tendon reattachment or graft insertion would induce unwanted changes in moment arms, whose magnitudes may be different depending on the direction of positional shift. The information of sensitivity could help the surgeons or manufacturers of medical devices to focus on limiting the error in the direction with potentially larger moment arm variance, so as not to compromise the biomechanical function of the operated shoulder.

Finally, we recapitulate the universal practicality of the concept we employed in the derivation of optimization gradients. A complex function may be disassembled into a product of simple modules for separate derivation, and the gradients of each module not only can be reassembled into the desired gradient composite, but may also be used as part of the gradient in other cost functions. A good example would be $\partial\boldsymbol{u}/\partial\boldsymbol{p}$ in (4), from which we conveniently derived the gradient for a path coordinate–based cost function. This enables calibration with medical imaging data (e.g., MRI) to expand the applicability. Furthermore, it would be interesting if kinetic measurements such as joint moment can be utilized in calibration, which requires the optimization of musculotendon parameters. In a previous study, we have analytically derived the requisite gradient [12], and it can be integrated with this work, making possible joint moment–based model calibration, which is key to automated musculoskeletal modeling.

V. Conclusion

A gradient-based method is developed for automated muscle path calibration, where path-related parameters are optimized to minimize the error in muscle moment arm. This method features specifying the gradient in its analytical form, which enables efficient computation and rapid convergence compared with using the numerical gradient, and the performance is demonstrated by fast and accurate replication of paths from a 12-DoF 42-muscle human shoulder–arm model.

We explain the derivation in detail, which first requires to smooth the cost function by replacing the conditional statements and nondifferentiable components with soft functions, and then modularize it into multiplicative factors that are easier to derive separately. This concept applies to other cost functions as well, and should be practical in the optimization of various properties in musculoskeletal models as well as many other systems.

References

[1] B. A. Garner and M. G. Pandy, "The obstacle-set method for representing muscle paths in musculoskeletal models," *Computer Methods in Biomechanics and Biomedical Engineering*, vol. 3, no. 1, pp. 1–30, 2000.

[2] J. L. Hicks *et al.*, "Is my model good enough? Best practices for verification and validation of musculoskeletal models and simulations of movement," *Journal of Biomechanical Engineering*, vol. 137, no. 2, p. 020905, 2015.

[3] J. Guo *et al.*, "Modeling muscle wrapping and mass flow using a mass-variable multibody formulation," *Multibody System Dynamics*, vol. 49, no. 3, pp. 315–336, 2020.

[4] J. E. Lloyd, F. Roewer-Després, and I. Stavness, "Muscle path wrapping on arbitrary surfaces," *IEEE Transactions on Biomedical Engineering*, vol. 68, no. 2, pp. 628–638, 2021.

[5] S. L. Delp, "Surgery simulation: A computer graphics system to analyze and design musculoskeletal reconstructions of the lower limb," Doctoral dissertation, Stanford University, 1990.

[6] K. R. S. Holzbaur, W. M. Murray, and S. L. Delp, "A model of the upper extremity for simulating musculoskeletal surgery and analyzing neuromuscular control," *Annals of Biomedical Engineering*, vol. 33, no. 6, pp. 829–840, 2005.

[7] E. M. Arnold *et al.*, "A model of the lower limb for analysis of human movement," *Annals of Biomedical Engineering*, vol. 38, no. 2, pp. 269–279, 2010.

[8] J. Ma'touq, T. Hu, and S. Haddadin, "A validated combined musculotendon path and muscle-joint kinematics model for the human hand," *Computer Methods in Biomechanics and Biomedical Engineering*, vol. 22, no. 7, pp. 727–739, 2019.

[9] A. M. Gordon, A. F. Huxley, and F. J. Julian, "The variation in isometric tension with sarcomere length in vertebrate muscle fibres," *The Journal of Physiology*, vol. 184, no. 1, pp. 170–192, 1966.

[10] S. K. Gollapudi and D. C. Lin, "Experimental determination of sarcomere force–length relationship in type-I human skeletal muscle fibers," *Journal of Biomechanics*, vol. 42, no. 13, pp. 2011–2016, 2009.

[11] M. Millard *et al.*, "Flexing computational muscle: Modeling and simulation of musculotendon dynamics," *Journal of Biomechanical Engineering*, vol. 135, no. 2, p. 021005, 2013.

[12] Z. Chen and D. W. Franklin, "Musculotendon parameters in lower limb models: Simplifications, uncertainties, and muscle force estimation sensitivity," *Annals of Biomedical Engineering*, vol. 51, no. 6, pp. 1147–1164, 2023.

[13] M. Millard, D. W. Franklin, and W. Herzog, "A three filament mechanistic model of musculotendon force and impedance," *eLife*, vol. 12, p. RP88344, 2024.

[14] K. N. An *et al.*, "Determination of muscle orientations and moment arms," *Journal of Biomechanical Engineering*, vol. 106, no. 3, pp. 280–282, 1984.

[15] J. N. Weinstein, T. P. Andriacchi, and J. O. Galante, "A relationship of total knee design kinematics to patient function," *The Iowa Orthopaedic Journal*, vol. 7, pp. 85–90, 1987.

[16] R. L. Aper, C. L. Saltzman, and T. D. Brown, "The effect of hallux sesamoid excision on the flexor hallucis longus moment arm," *Clinical Orthopaedics and Related Research*, vol. 325, pp. 209–217, 1996.

[17] A. S. Arnold *et al.*, "Accuracy of muscle moment arms estimated from MRI-based musculoskeletal models of the lower extremity," *Computer Aided Surgery*, vol. 5, no. 2, pp. 108–119, 2000.

[18] B. A. Garner and M. G. Pandy, "Musculoskeletal model of the upper limb based on the Visible Human Male dataset," *Computer Methods in Biomechanics and Biomedical Engineering*, vol. 4, no. 2, pp. 93–126, 2001.

[19] L. Scheys *et al.*, "Calculated moment-arm and muscle–tendon lengths during gait differ substantially using MR based versus rescaled generic lower-limb musculoskeletal models," *Gait & Posture*, vol. 28, no. 4, pp. 640–648, 2008.

[20] B. A. Killen *et al.*, "Automated creation and tuning of personalised muscle paths for OpenSim musculoskeletal models of the knee joint," *Biomechanics and Modeling in Mechanobiology*, vol. 20, no. 2, pp. 521–533, 2021.

[21] D. Ao *et al.*, "EMG-driven musculoskeletal model calibration with wrapping surface personalization," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 31, pp. 4235–4244, 2023.

[22] S. L. Delp *et al.*, "OpenSim: Open-source software to create and analyze dynamic simulations of movement," *IEEE Transactions on Biomedical Engineering*, vol. 54, no. 11, pp. 1940–1950, 2007.

[23] L. Modenese and J. Kohout, "Automated generation of three-dimensional complex muscle geometries for use in personalised musculoskeletal models," *Annals of Biomedical Engineering*, vol. 48, no. 6, pp. 1793–1804, 2020.

[24] D. C. McFarland *et al.*, "A musculoskeletal model of the hand and wrist capable of simulating functional tasks," *IEEE Transactions on Biomedical Engineering*, vol. 70, no. 5, pp. 1424–1435, 2023.

[25] A. Rajagopal *et al.*, "Full-body musculoskeletal model for muscle-driven simulation of human gait," *IEEE Transactions on Biomedical Engineering*, vol. 63, no. 10, pp. 2068–2079, 2016.

[26] P. Wretenberg *et al.*, "Passive knee muscle moment arms measured in vivo with MRI," *Clinical Biomechanics*, vol. 11, no. 8, pp. 439–446, 1996.

[27] M. B. McCullough *et al.*, "Moment arms of the ankle throughout the range of motion in three planes," *Foot & Ankle International*, vol. 32, no. 3, pp. 300–306, 2011.

[28] K. Vaarbakken *et al.*, "Lengths of the external hip rotators in mobilized cadavers indicate the quadriceps coxa as a primary abductor and extensor of the flexed hip," *Clinical Biomechanics*, vol. 29, no. 7, pp. 794–802, 2014.

[29] C. N. Maganaris, V. Baltzopoulos, and A. J. Sargeant, "In vivo measurement-based estimations of the human Achilles tendon moment arm," *European Journal of Applied Physiology*, vol. 83, no. 4-5, pp. 363–369, 2000.

[30] S. R. Ward *et al.*, "Are current measurements of lower extremity muscle architecture accurate?" *Clinical Orthopaedics and Related Research*, vol. 467, no. 4, pp. 1074–1082, 2009.

[31] F. Fath *et al.*, "Direct comparison of in vivo Achilles tendon moment arms obtained from ultrasound and MR scans," *Journal of Applied Physiology*, vol. 109, no. 6, pp. 1644–1652, 2010.

[32] B. Hintermann, B. M. Nigg, and C. Sommer, "Foot movement and tendon excursion: An in vitro study," *Foot & Ankle International*, vol. 15, no. 7, pp. 386–395, 1994.

[33] S. S. Lee and S. J. Piazza, "Inversion–eversion moment arms of gastrocnemius and tibialis anterior measured in vivo," *Journal of Biomechanics*, vol. 41, no. 16, pp. 3366–3370, 2008.

[34] W. Herzog and H. E. D. J. ter Keurs, "A method for the determination of the force–length relation of selected in-vivo human skeletal muscles," *Pflügers Archiv*, vol. 411, no. 6, pp. 637–641, 1988.

[35] S. L. Delp *et al.*, "Variation of rotation moment arms with hip flexion," *Journal of Biomechanics*, vol. 32, no. 5, pp. 493–501, 1999.

[36] S. Wolfram *et al.*, "Achilles tendon moment arm in humans is not affected by inversion/eversion of the foot: A short report," *Royal Society Open Science*, vol. 5, no. 1, p. 171358, 2018.

[37] T. R. Kane and D. A. Levinson, *Dynamics: Theory and applications*. McGraw-Hill, 1985.

[38] G. T. Yamaguchi, *Dynamic modeling of musculoskeletal motion: A vectorized approach for biomechanical analysis in three dimensions*. Springer US, 2001.

[39] M. A. Sherman, A. Seth, and S. L. Delp, "What is a moment arm? Calculating muscle effectiveness in biomechanical models using generalized coordinates," in *9th International Conference on Multibody Systems, Nonlinear Dynamics, and Control*. American Society of Mechanical Engineers, 2013, p. V07BT10A052.

[40] T. Hu, J. Kühn, and S. Haddadin, "Forward and inverse dynamics modeling of human shoulder–arm musculoskeletal system with scapulothoracic constraint," *Computer Methods in Biomechanics and Biomedical Engineering*, vol. 23, no. 11, pp. 785–803, 2020.

[41] T. Hu, J. Kuehn, and S. Haddadin, "Identification of human shoulder-arm kinematic and muscular synergies during daily-life manipulation tasks," in *2018 7th IEEE International Conference on Biomedical Robotics and Biomechatronics*. IEEE, 2018, pp. 1011–1018.

[42] J. Nocedal and S. J. Wright, *Numerical optimization*. Springer New York, 2006.

[43] The MathWorks Inc., "Least-squares (model fitting) algorithms documentation," The MathWorks Inc., 2024. [Online]. Available: https://www.mathworks.com/help/optim/ug/least-squares-model-fitting-algorithms.html

[44] J. E. Hopcroft and D. Rus, "Algorithms, analysis of," in *Encyclopedia of Computer Science*. John Wiley and Sons Ltd., 2003, pp. 42–45.

[45] C. H. Papadimitriou, "Computational complexity," in *Encyclopedia of Computer Science*. John Wiley and Sons Ltd., 2003, pp. 260–265.

[46] P. S. Oliveto, J. He, and X. Yao, "Time complexity of evolutionary algorithms for combinatorial optimization: A decade of results," *International Journal of Automation and Computing*, vol. 4, no. 3, pp. 281–293, 2007.

[47] S. Forrest, "Genetic algorithms: Principles of natural selection applied to computation," *Science*, vol. 261, no. 5123, pp. 872–878, 1993.

[48] S. Mirjalili, "Genetic Algorithm," in *Evolutionary Algorithms and Neural Networks: Theory and Applications*. Springer International Publishing, 2019, pp. 43–55.

[49] D. Holzer *et al.*, "Considerations on the human Achilles tendon moment arm for in vivo triceps surae muscle–tendon unit force estimates," *Scientific Reports*, vol. 10, no. 1, p. 19559, 2020.

[50] Z. Chen and D. W. Franklin, "Muscle moment arm–joint angle relations in the hip, knee, and ankle: A visualization of datasets," *Annals of Biomedical Engineering*, 2025.

APPENDIX

A. Coordinate Transformation and Partial Angular Velocity

In our mathematical derivation, there are three types of reference frames in which the coordinates of a point are expressed: world (w), joint (j), and cylinder (c), and the transformations in-between are proceeded as

$$\begin{bmatrix} {}^{\mathrm{w}}\boldsymbol{u} \\ 1 \end{bmatrix} = {}^{\mathrm{w}}\boldsymbol{T}_{\mathrm{j}} \begin{bmatrix} {}^{\mathrm{j}}\boldsymbol{u} \\ 1 \end{bmatrix} = {}^{\mathrm{w}}\boldsymbol{T}_{\mathrm{j}}\, {}^{\mathrm{j}}\boldsymbol{T}_{\mathrm{c}} \begin{bmatrix} {}^{\mathrm{c}}\boldsymbol{u} \\ 1 \end{bmatrix}, \tag{S1}$$

where ${}^{\mathrm{w}}\boldsymbol{T}_{\mathrm{j}}$ is the transformation matrix from some specific joint frame to the world frame, and ${}^{\mathrm{j}}\boldsymbol{T}_{\mathrm{c}}$ cylinder to joint. Notice the anchor points and the center(s) of the cylinder(s) may be defined with respect to different joint frames, and unless specified, j refers to the correspondent frame in which a point is fixed against joint rotation; that is, the implication of j may be different between ${}^{\mathrm{j}}\boldsymbol{u}_{\mathrm{P}}$, ${}^{\mathrm{j}}\boldsymbol{u}_{\mathrm{S}}$, and ${}^{\mathrm{j}}\boldsymbol{u}_{\mathrm{C}}$.

For a path segment, the computation of moment arm requires the *partial angular velocity* from the anchor points and the valid wrapping points, and it is expressed as, e.g.,

$$\begin{aligned} {}^{\mathrm{jc}}\boldsymbol{\omega}_{\mathrm{Q}} &= \begin{bmatrix} \frac{\partial\, {}^{\mathrm{w}}\boldsymbol{T}^{*}_{\mathrm{jc}}}{\partial q_1} \begin{bmatrix} {}^{\mathrm{jc}}\boldsymbol{u}_{\mathrm{Q}} \\ 1 \end{bmatrix} & \frac{\partial\, {}^{\mathrm{w}}\boldsymbol{T}^{*}_{\mathrm{jc}}}{\partial q_2} \begin{bmatrix} {}^{\mathrm{jc}}\boldsymbol{u}_{\mathrm{Q}} \\ 1 \end{bmatrix} & \cdots \end{bmatrix} \\ &= \begin{bmatrix} \frac{\partial\, {}^{\mathrm{w}}\boldsymbol{T}^{*}_{\mathrm{jc}}}{\partial q_1}\, {}^{\mathrm{jc}}\boldsymbol{T}_{\mathrm{c}} \begin{bmatrix} {}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{Q}} \\ 1 \end{bmatrix} & \frac{\partial\, {}^{\mathrm{w}}\boldsymbol{T}^{*}_{\mathrm{jc}}}{\partial q_2}\, {}^{\mathrm{jc}}\boldsymbol{T}_{\mathrm{c}} \begin{bmatrix} {}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{Q}} \\ 1 \end{bmatrix} & \cdots \end{bmatrix}, \end{aligned} \tag{S2}$$

where the asterisk denotes the exclusion of the last row in the transformation matrix. The superscript refers to the joint frame in which the coordinates of the point (designated by the subscript) are expressed. Importantly, the designated point does not need to be fixed with respect to this joint frame.

B. Verification of Smooth Modification

The smooth modification of the obstacle-set method [1] and moment arm computation is verified in a way similar to the validation process described in Subsection II-C:

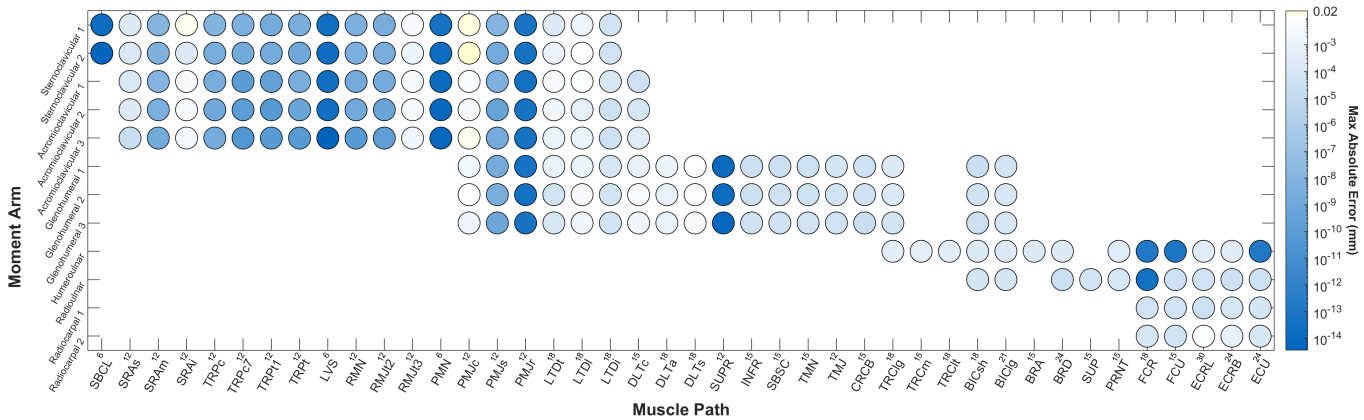
$$\epsilon_{\text{verAbs},i} = \max\nolimits_{n=1}^{1000} |r_{\text{modelGP},i}(\boldsymbol{q}_n, \boldsymbol{p}_{\text{ref}}) - r_{\text{modelGP+},i}(\boldsymbol{q}_n, \boldsymbol{p}_{\text{ref}})|, \tag{S3}$$

and

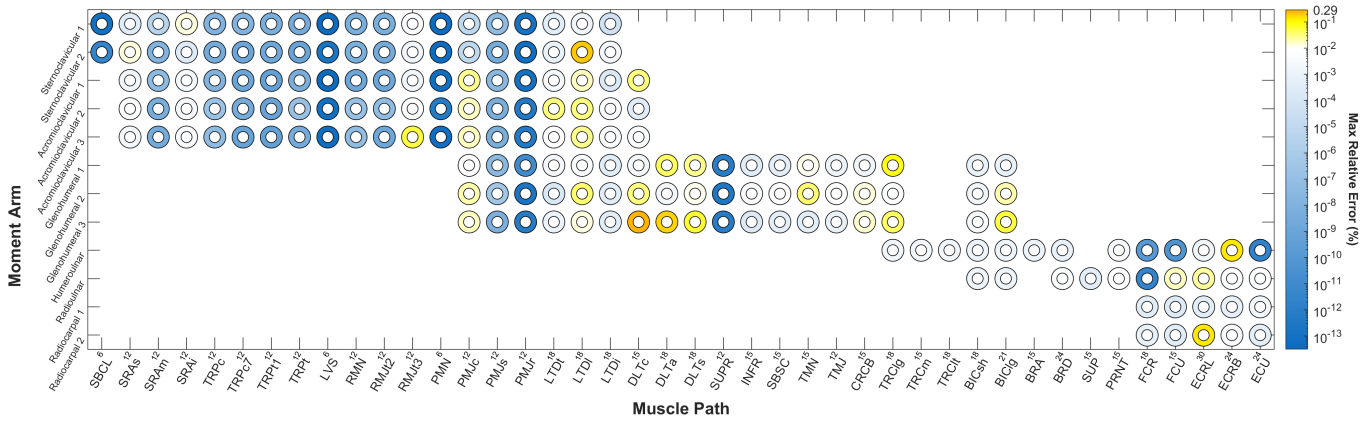
$$\begin{aligned} \epsilon_{\text{verRel},i} &= \max\nolimits_{n \in N^*} \left| \frac{r_{\text{modelGP},i}(\boldsymbol{q}_n, \boldsymbol{p}_{\text{ref}}) - r_{\text{modelGP+},i}(\boldsymbol{q}_n, \boldsymbol{p}_{\text{ref}})}{r_{\text{modelGP},i}(\boldsymbol{q}_n, \boldsymbol{p}_{\text{ref}})} \right|, \\ N^* &= \{n \mid 1 \le n \le 1000, r_{\text{modelGP},i}(\boldsymbol{q}_n, \boldsymbol{p}_{\text{ref}}) \ge 0.01\}, \end{aligned} \tag{S4}$$

where $r_{\text{modelGP},i}$ is the i-th moment arm computed with the slightly revised obstacle-set method (Fig. 1, middle) and $\boldsymbol{r}_{\text{modelGP+},i}$ with the smooth form described in Subsection II-B. The input $\boldsymbol{p}_{\text{ref}}$ is path-related parameters from [18], and $\boldsymbol{q}_n$ is interpolated from the kinematic measurements of the shoulder and arm from 10 intransitive daily tasks (100 joint configurations per task) performed by a single subject in [41], including gesturing an OK sign, pumping fists, blocking out light from the face, saluting, pointing, gesturing a thumb-down, signaling for hitchhike, greeting, gesturing to stop, and gesturing for silence. For $\epsilon_{\text{verRel},i}$, a joint configuration will be excluded if the correspondent $|r_{\text{modelGP},i}|$ is below 0.01 mm (indicated by N^*).

Moment arm is computed in a 12-DoF human shoulder–arm model [40]; see Supplementary Table I for DoFs in detail. As shown in the figures below, across 1000 joint configurations in the range of motion, the moment arm computed based on our smoothed method has negligible difference from the original value, with the maximum from 182 effective moment arms of 42 muscle paths being 0.02 mm; see Supplementary Table II for muscle paths in detail. The existing difference stems from the nature of the soft functions, rather than coding issues.



S Fig. 1. Absolute errors in 182 effective moment arms of 42 muscle paths (max value across 1000 joint configurations). The magnitude of the error is indicated by the intensity of the color. The number in the horizontal axis corresponds to the parameter number for each muscle. See Appendix D for nomenclatures in the axes.



S Fig. 2. Relative errors in 182 effective moment arms of 42 muscle paths (max value across all valid joint configurations). The magnitude of the relative percentage is indicated by the intensity of the color. The number in the horizontal axis corresponds to the parameter number for each muscle. See Appendix D for nomenclatures in the axes.

C. Derivation of Gradients

In our mathematical derivation, the gradient of a scalar variable is formatted as a row vector, while that of a vector variable is formatted as a matrix, namely the Jacobian, whose rows are the gradients of each scalar variable. Suppose for the single-cylinder case with 12 parameters, $\partial R/\partial \boldsymbol{p}$ is simply a 12-D row vector whose seventh element is 1 while the rest are 0, and $\partial\, {}^{\mathrm{j}}\boldsymbol{u}_{\mathrm{P}}/\partial \boldsymbol{p}$ is a 3×3 identity matrix concatenated with a 3×9 null matrix.

In Step 1, the coordinates of the anchor points are transformed from the joint frame to the cylinder frame, e.g.,

$$\begin{bmatrix} {}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{P}} \\ 1 \end{bmatrix} = {}^{\mathrm{c}}\boldsymbol{T}_{\mathrm{j_C}}({}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{C}}, \alpha, \beta)\, {}^{\mathrm{j_C}}\boldsymbol{T}_{\mathrm{j_P}} \begin{bmatrix} {}^{\mathrm{j}}\boldsymbol{u}_{\mathrm{P}} \\ 1 \end{bmatrix}, \tag{S5}$$

where ${}^{\mathrm{j_C}}\boldsymbol{T}_{\mathrm{j_P}}$ is the transformation matrix between two joint frames (dependent on $\boldsymbol{q}$ and irrelevant to $\boldsymbol{p}$), and ${}^{\mathrm{c}}\boldsymbol{T}_{\mathrm{j_C}}({}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{C}}, \alpha, \beta)$ is the transformation matrix from the joint frame to the cylinder frame, determined by the location and orientation of the cylinder

$${}^{\mathrm{c}}\boldsymbol{T}_{\mathrm{j_C}} = {}^{\mathrm{j_C}}\boldsymbol{T}_{\mathrm{c}}{}^{-1} = \left[\begin{array}{ccc:c} \cos\beta & 0 & \sin\beta & {}^{\mathrm{j}}x_{\mathrm{C}} \\ \sin\alpha\sin\beta & \cos\alpha & -\cos\beta\sin\alpha & {}^{\mathrm{j}}y_{\mathrm{C}} \\ -\cos\alpha\sin\beta & \sin\alpha & \cos\alpha\cos\beta & {}^{\mathrm{j}}z_{\mathrm{C}} \\ \hdashline 0 & 0 & 0 & 1 \end{array}\right]^{-1}. \tag{S6}$$

Based on the product rule, we may then easily compute the gradient, e.g.,

$$\begin{aligned}\underbrace{\begin{bmatrix}\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{P}}/\partial\boldsymbol{p}\\ 0\ldots0\end{bmatrix}}_{4\times12} &= \Bigg[\underbrace{\frac{\partial\,{}^{\mathrm{c}}\boldsymbol{T}_{\mathrm{j_C}}}{\partial\,{}^{\mathrm{j_C}}x_{\mathrm{P}}}\begin{bmatrix}{}^{\mathrm{j_C}}\boldsymbol{u}_{\mathrm{P}}\\1\end{bmatrix}\cdots\frac{\partial\,{}^{\mathrm{c}}\boldsymbol{T}_{\mathrm{j_C}}}{\partial\,{}^{\mathrm{j_C}}z_{\mathrm{S}}}\begin{bmatrix}{}^{\mathrm{j_C}}\boldsymbol{u}_{\mathrm{P}}\\1\end{bmatrix}}_{4\times6}\ \vdots\ \underbrace{\frac{\partial\,{}^{\mathrm{c}}\boldsymbol{T}_{\mathrm{j_C}}}{\partial R}\begin{bmatrix}{}^{\mathrm{j_C}}\boldsymbol{u}_{\mathrm{P}}\\1\end{bmatrix}}_{4\times1}\ \vdots\ \underbrace{\frac{\partial\,{}^{\mathrm{c}}\boldsymbol{T}_{\mathrm{j_C}}}{\partial\,{}^{\mathrm{j}}x_{\mathrm{C}}}\begin{bmatrix}{}^{\mathrm{j_C}}\boldsymbol{u}_{\mathrm{P}}\\1\end{bmatrix}\cdots}_{4\times3}\ \vdots\ \underbrace{\frac{\partial\,{}^{\mathrm{c}}\boldsymbol{T}_{\mathrm{j_C}}}{\partial\alpha}\begin{bmatrix}{}^{\mathrm{j_C}}\boldsymbol{u}_{\mathrm{P}}\\1\end{bmatrix}\cdots}_{4\times2}\Bigg]\\ &\qquad+{}^{\mathrm{c}}\boldsymbol{T}_{\mathrm{j_C}}\underbrace{\begin{bmatrix}\partial\,{}^{\mathrm{j_C}}\boldsymbol{u}_{\mathrm{P}}/\partial\boldsymbol{p}\\0\ldots0\end{bmatrix}}_{4\times12}\\ &= \Bigg[\underset{4\times6}{\mathbf{O}}\ \vdots\ \underset{4\times1}{\mathbf{O}}\ \vdots\ \underbrace{\frac{\partial\,{}^{\mathrm{c}}\boldsymbol{T}_{\mathrm{j_C}}}{\partial\,{}^{\mathrm{j}}x_{\mathrm{C}}}\begin{bmatrix}{}^{\mathrm{j_C}}\boldsymbol{u}_{\mathrm{P}}\\1\end{bmatrix}\quad\frac{\partial\,{}^{\mathrm{c}}\boldsymbol{T}_{\mathrm{j_C}}}{\partial\,{}^{\mathrm{j}}y_{\mathrm{C}}}\begin{bmatrix}{}^{\mathrm{j_C}}\boldsymbol{u}_{\mathrm{P}}\\1\end{bmatrix}\quad\frac{\partial\,{}^{\mathrm{c}}\boldsymbol{T}_{\mathrm{j_C}}}{\partial\,{}^{\mathrm{j}}z_{\mathrm{C}}}\begin{bmatrix}{}^{\mathrm{j_C}}\boldsymbol{u}_{\mathrm{P}}\\1\end{bmatrix}}_{4\times3}\ \vdots\ \underbrace{\frac{\partial\,{}^{\mathrm{c}}\boldsymbol{T}_{\mathrm{j_C}}}{\partial\alpha}\begin{bmatrix}{}^{\mathrm{j_C}}\boldsymbol{u}_{\mathrm{P}}\\1\end{bmatrix}\quad\frac{\partial\,{}^{\mathrm{c}}\boldsymbol{T}_{\mathrm{j_C}}}{\partial\beta}\begin{bmatrix}{}^{\mathrm{j_C}}\boldsymbol{u}_{\mathrm{P}}\\1\end{bmatrix}}_{4\times2}\Bigg]\\ &\qquad+{}^{\mathrm{c}}\boldsymbol{T}_{\mathrm{j_C}}\begin{bmatrix}\underset{3\times3}{\mathbf{I}} & \underset{3\times9}{\mathbf{O}}\\ 0\ \ldots\ 0 & \end{bmatrix},\end{aligned}\tag{S7}$$

and notice the partial derivatives with respect to $\boldsymbol{p}$ are in the order of ${}^{\mathrm{j}}x_{\mathrm{P}}$, ${}^{\mathrm{j}}y_{\mathrm{P}}$, ${}^{\mathrm{j}}z_{\mathrm{P}}$, ${}^{\mathrm{j}}x_{\mathrm{S}}$, ${}^{\mathrm{j}}y_{\mathrm{S}}$, ${}^{\mathrm{j}}z_{\mathrm{S}}$, R, ${}^{\mathrm{j}}x_{\mathrm{C}}$, ${}^{\mathrm{j}}y_{\mathrm{C}}$, ${}^{\mathrm{j}}z_{\mathrm{C}}$, α, and β.

Step 1.5 checks if the anchor points are outside the cylinder, and we utilized (8) to return adjusted coordinates as well as their gradients, e.g.,

$${}^{\mathrm{c}}x_{\tilde{\mathrm{P}}} = {}^{\mathrm{c}}x_{\mathrm{P}'} + \left({}^{\mathrm{c}}x_{\mathrm{P}} - {}^{\mathrm{c}}x_{\mathrm{P}'}\right) f_{\text{softStep}}(\delta_{\mathrm{P}}),\tag{S8}$$

$$\begin{aligned}\frac{\partial\,{}^{\mathrm{c}}x_{\tilde{\mathrm{P}}}}{\partial\boldsymbol{p}} &= \frac{{}^{\mathrm{c}}x_{\mathrm{P}'}}{\partial\boldsymbol{p}} + \frac{\partial\left({}^{\mathrm{c}}x_{\mathrm{P}} - {}^{\mathrm{c}}x_{\mathrm{P}'}\right)}{\partial\boldsymbol{p}} f_{\text{softStep}}(\delta_{\mathrm{P}})\\ &\qquad+\left({}^{\mathrm{c}}x_{\mathrm{P}} - {}^{\mathrm{c}}x_{\mathrm{P}'}\right)\frac{\partial f_{\text{softStep}}(\delta_{\mathrm{P}})}{\partial\boldsymbol{p}},\end{aligned}\tag{S9}$$

where all of the partial derivative terms can be easily derived using the chain rule. For instance,

$$\begin{aligned}\frac{\partial f_{\text{softStep}}(\delta_{\mathrm{P}})}{\partial\boldsymbol{p}} &= \frac{\partial f_{\text{softStep}}}{\partial\delta_{\mathrm{P}}}\ \Bigg[\underbrace{\frac{\partial\delta_{\mathrm{P}}}{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{P}}}}_{1\times3}\ \vdots\ \frac{\partial\delta_{\mathrm{P}}}{\partial R}\Bigg]\ \underbrace{\begin{bmatrix}\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{P}}/\partial\boldsymbol{p}\\ \hdashline \partial R/\partial\boldsymbol{p}\end{bmatrix}}_{4\times12}\\ &= \frac{2}{(e^{-2\delta_{\mathrm{P}}}+1)(e^{2\delta_{\mathrm{P}}}+1)}\ 2\ \Big[{}^{\mathrm{c}}x_{\mathrm{P}}\ \ {}^{\mathrm{c}}y_{\mathrm{P}}\ \ 0\ \vdots\ -R\Big]\ \begin{bmatrix}\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{P}}/\partial\boldsymbol{p}\\ \hdashline \underset{1\times6}{\mathbf{O}}\ \ 1\ \ \underset{1\times5}{\mathbf{O}}\end{bmatrix},\end{aligned}\tag{S10}$$

where $\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{P}}/\partial\boldsymbol{p}$ is the only variable unspecified in $\boldsymbol{p}$ and may be obtained from (S7). To avoid inflating the equations, we omit the process of expanding each and every partial derivative terms in the rest of this section and focus on the format instead.

Next, with ${}^{\mathrm{c}}x_{\tilde{\mathrm{P}}}$, ${}^{\mathrm{c}}y_{\tilde{\mathrm{P}}}$, ${}^{\mathrm{c}}x_{\tilde{\mathrm{S}}}$, and ${}^{\mathrm{c}}y_{\tilde{\mathrm{S}}}$ from (S8), we can derive the x and y coordinates of the wrapping points as described in [1]:

$$\begin{aligned}\begin{bmatrix}{}^{\mathrm{c}}x_{\mathrm{Q}}\\{}^{\mathrm{c}}y_{\mathrm{Q}}\end{bmatrix} &= \begin{bmatrix}\dfrac{{}^{\mathrm{c}}x_{\tilde{\mathrm{P}}}R^2 - {}^{\mathrm{c}}y_{\tilde{\mathrm{P}}}R\sqrt{{}^{\mathrm{c}}x_{\tilde{\mathrm{P}}}{}^2+{}^{\mathrm{c}}y_{\tilde{\mathrm{P}}}{}^2-R^2}}{{}^{\mathrm{c}}x_{\tilde{\mathrm{P}}}{}^2+{}^{\mathrm{c}}y_{\tilde{\mathrm{P}}}{}^2}\\ \dfrac{{}^{\mathrm{c}}y_{\tilde{\mathrm{P}}}R^2 + {}^{\mathrm{c}}x_{\tilde{\mathrm{P}}}R\sqrt{{}^{\mathrm{c}}x_{\tilde{\mathrm{P}}}{}^2+{}^{\mathrm{c}}y_{\tilde{\mathrm{P}}}{}^2-R^2}}{{}^{\mathrm{c}}x_{\tilde{\mathrm{P}}}{}^2+{}^{\mathrm{c}}y_{\tilde{\mathrm{P}}}{}^2}\end{bmatrix},\\ \begin{bmatrix}{}^{\mathrm{c}}x_{\mathrm{T}}\\{}^{\mathrm{c}}y_{\mathrm{T}}\end{bmatrix} &= \begin{bmatrix}\dfrac{{}^{\mathrm{c}}x_{\tilde{\mathrm{S}}}R^2 + {}^{\mathrm{c}}y_{\tilde{\mathrm{S}}}R\sqrt{{}^{\mathrm{c}}x_{\tilde{\mathrm{S}}}{}^2+{}^{\mathrm{c}}y_{\tilde{\mathrm{S}}}{}^2-R^2}}{{}^{\mathrm{c}}x_{\tilde{\mathrm{S}}}{}^2+{}^{\mathrm{c}}y_{\tilde{\mathrm{S}}}{}^2}\\ \dfrac{{}^{\mathrm{c}}y_{\tilde{\mathrm{S}}}R^2 - {}^{\mathrm{c}}x_{\tilde{\mathrm{S}}}R\sqrt{{}^{\mathrm{c}}x_{\tilde{\mathrm{S}}}{}^2+{}^{\mathrm{c}}y_{\tilde{\mathrm{S}}}{}^2-R^2}}{{}^{\mathrm{c}}x_{\tilde{\mathrm{S}}}{}^2+{}^{\mathrm{c}}y_{\tilde{\mathrm{S}}}{}^2}\end{bmatrix},\end{aligned}\tag{S11}$$

and additionally with the respective partial derivatives from (S9), we can also compute their gradients

$$\begin{aligned}\frac{\partial\,{}^{\mathrm{c}}x_{\mathrm{Q}}}{\partial\boldsymbol{p}} &= \begin{bmatrix}\frac{\partial\,{}^{\mathrm{c}}x_{\mathrm{Q}}}{\partial\,{}^{\mathrm{c}}x_{\tilde{\mathrm{P}}}} & \frac{\partial\,{}^{\mathrm{c}}x_{\mathrm{Q}}}{\partial\,{}^{\mathrm{c}}y_{\tilde{\mathrm{P}}}} & \frac{\partial\,{}^{\mathrm{c}}x_{\mathrm{Q}}}{\partial R}\end{bmatrix}\begin{bmatrix}\frac{\partial\,{}^{\mathrm{c}}x_{\tilde{\mathrm{P}}}}{\partial\boldsymbol{p}} & \frac{\partial\,{}^{\mathrm{c}}y_{\tilde{\mathrm{P}}}}{\partial\boldsymbol{p}} & \frac{\partial R}{\partial\boldsymbol{p}}\end{bmatrix}^{\mathrm{T}},\\ \frac{\partial\,{}^{\mathrm{c}}y_{\mathrm{Q}}}{\partial\boldsymbol{p}} &= \begin{bmatrix}\frac{\partial\,{}^{\mathrm{c}}y_{\mathrm{Q}}}{\partial\,{}^{\mathrm{c}}x_{\tilde{\mathrm{P}}}} & \frac{\partial\,{}^{\mathrm{c}}y_{\mathrm{Q}}}{\partial\,{}^{\mathrm{c}}y_{\tilde{\mathrm{P}}}} & \frac{\partial\,{}^{\mathrm{c}}y_{\mathrm{Q}}}{\partial R}\end{bmatrix}\begin{bmatrix}\frac{\partial\,{}^{\mathrm{c}}x_{\tilde{\mathrm{P}}}}{\partial\boldsymbol{p}} & \frac{\partial\,{}^{\mathrm{c}}y_{\tilde{\mathrm{P}}}}{\partial\boldsymbol{p}} & \frac{\partial R}{\partial\boldsymbol{p}}\end{bmatrix}^{\mathrm{T}},\\ \frac{\partial\,{}^{\mathrm{c}}x_{\mathrm{T}}}{\partial\boldsymbol{p}} &= \begin{bmatrix}\frac{\partial\,{}^{\mathrm{c}}x_{\mathrm{T}}}{\partial\,{}^{\mathrm{c}}x_{\tilde{\mathrm{S}}}} & \frac{\partial\,{}^{\mathrm{c}}x_{\mathrm{T}}}{\partial\,{}^{\mathrm{c}}y_{\tilde{\mathrm{S}}}} & \frac{\partial\,{}^{\mathrm{c}}x_{\mathrm{T}}}{\partial R}\end{bmatrix}\begin{bmatrix}\frac{\partial\,{}^{\mathrm{c}}x_{\tilde{\mathrm{S}}}}{\partial\boldsymbol{p}} & \frac{\partial\,{}^{\mathrm{c}}y_{\tilde{\mathrm{S}}}}{\partial\boldsymbol{p}} & \frac{\partial R}{\partial\boldsymbol{p}}\end{bmatrix}^{\mathrm{T}},\\ \frac{\partial\,{}^{\mathrm{c}}y_{\mathrm{T}}}{\partial\boldsymbol{p}} &= \begin{bmatrix}\frac{\partial\,{}^{\mathrm{c}}y_{\mathrm{T}}}{\partial\,{}^{\mathrm{c}}x_{\tilde{\mathrm{S}}}} & \frac{\partial\,{}^{\mathrm{c}}y_{\mathrm{T}}}{\partial\,{}^{\mathrm{c}}y_{\tilde{\mathrm{S}}}} & \frac{\partial\,{}^{\mathrm{c}}y_{\mathrm{T}}}{\partial R}\end{bmatrix}\begin{bmatrix}\frac{\partial\,{}^{\mathrm{c}}x_{\tilde{\mathrm{S}}}}{\partial\boldsymbol{p}} & \frac{\partial\,{}^{\mathrm{c}}y_{\tilde{\mathrm{S}}}}{\partial\boldsymbol{p}} & \frac{\partial R}{\partial\boldsymbol{p}}\end{bmatrix}^{\mathrm{T}},\end{aligned}\tag{S12}$$

where the first terms on the right side are easy to derive from (S11), and the second terms are obtained from (S7) and (S9).

Step 2 ends with the calculation of ${}^{\mathrm{c}}z_{\mathrm{Q}}$ and ${}^{\mathrm{c}}z_{\mathrm{T}}$ as described in [1]:

$$
\begin{aligned}
{}^{\mathrm{c}}z_{\mathrm{Q}} &= {}^{\mathrm{c}}z_{\tilde{\mathrm{P}}} + ({}^{\mathrm{c}}z_{\tilde{\mathrm{S}}} - {}^{\mathrm{c}}z_{\tilde{\mathrm{P}}}) \frac{f_{\mathrm{softNorm}}({}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{P}},\mathrm{xy}} - {}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{Q,xy}})}{f_{\mathrm{softNorm}}({}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{P}},\mathrm{xy}} - {}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{Q,xy}}) + f_{\mathrm{softNorm}}(R\gamma) + f_{\mathrm{softNorm}}({}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{T,xy}} - {}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{S}},\mathrm{xy}})}, \\
{}^{\mathrm{c}}z_{\mathrm{T}} &= {}^{\mathrm{c}}z_{\tilde{\mathrm{S}}} - ({}^{\mathrm{c}}z_{\tilde{\mathrm{S}}} - {}^{\mathrm{c}}z_{\tilde{\mathrm{P}}}) \frac{f_{\mathrm{softNorm}}({}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{T,xy}} - {}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{S}},\mathrm{xy}})}{f_{\mathrm{softNorm}}({}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{P}},\mathrm{xy}} - {}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{Q,xy}}) + f_{\mathrm{softNorm}}(R\gamma) + f_{\mathrm{softNorm}}({}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{T,xy}} - {}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{S}},\mathrm{xy}})}, \\
\gamma &= f_{\mathrm{softAtan2}}\Big(\frac{\boldsymbol{u}_{\mathrm{Q,xy}}^{\mathrm{T}} \boldsymbol{u}_{\mathrm{S,xy}}}{f_{\mathrm{softNorm}}({}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{Q,xy}}) f_{\mathrm{softNorm}}({}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{T,xy}})}, \frac{{}^{\mathrm{c}}x_{\mathrm{Q}}{}^{\mathrm{c}}y_{\mathrm{T}} - {}^{\mathrm{c}}x_{\mathrm{T}}{}^{\mathrm{c}}y_{\mathrm{Q}}}{f_{\mathrm{softNorm}}({}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{Q,xy}}) f_{\mathrm{softNorm}}({}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{T,xy}})} \Big),
\end{aligned}
\tag{S13}
$$

and Step 3 returns the coordinates of $\tilde{\mathrm{Q}}$ and $\tilde{\mathrm{T}}$ after checking the validity of the wrapping with (12). Likewise, their gradients are formatted as

$$
\begin{aligned}
\frac{\partial\, {}^{\mathrm{c}}z_{\mathrm{Q}}}{\partial \boldsymbol{p}} &= \begin{bmatrix} \frac{\partial\, {}^{\mathrm{c}}z_{\mathrm{Q}}}{\partial\, {}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{P}}}} & \frac{\partial\, {}^{\mathrm{c}}z_{\mathrm{Q}}}{\partial\, {}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{S}}}} & \frac{\partial\, {}^{\mathrm{c}}z_{\mathrm{Q}}}{\partial R} & \frac{\partial\, {}^{\mathrm{c}}z_{\mathrm{Q}}}{\partial\, {}^{\mathrm{c}}x_{\mathrm{Q}}} & \frac{\partial\, {}^{\mathrm{c}}z_{\mathrm{Q}}}{\partial\, {}^{\mathrm{c}}y_{\mathrm{Q}}} \end{bmatrix} \begin{bmatrix} \frac{\partial\, {}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{P}}}}{\partial \boldsymbol{p}} & \frac{\partial\, {}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{S}}}}{\partial \boldsymbol{p}} & \frac{\partial R}{\partial \boldsymbol{p}} & \frac{\partial\, {}^{\mathrm{c}}x_{\mathrm{Q}}}{\partial \boldsymbol{p}} & \frac{\partial\, {}^{\mathrm{c}}y_{\mathrm{Q}}}{\partial \boldsymbol{p}} \end{bmatrix}^{\mathrm{T}}, \\
\frac{\partial\, {}^{\mathrm{c}}z_{\mathrm{T}}}{\partial \boldsymbol{p}} &= \begin{bmatrix} \frac{\partial\, {}^{\mathrm{c}}z_{\mathrm{T}}}{\partial\, {}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{P}}}} & \frac{\partial\, {}^{\mathrm{c}}z_{\mathrm{T}}}{\partial\, {}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{S}}}} & \frac{\partial\, {}^{\mathrm{c}}z_{\mathrm{T}}}{\partial R} & \frac{\partial\, {}^{\mathrm{c}}z_{\mathrm{T}}}{\partial\, {}^{\mathrm{c}}x_{\mathrm{T}}} & \frac{\partial\, {}^{\mathrm{c}}z_{\mathrm{T}}}{\partial\, {}^{\mathrm{c}}y_{\mathrm{T}}} \end{bmatrix} \begin{bmatrix} \frac{\partial\, {}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{P}}}}{\partial \boldsymbol{p}} & \frac{\partial\, {}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{S}}}}{\partial \boldsymbol{p}} & \frac{\partial R}{\partial \boldsymbol{p}} & \frac{\partial\, {}^{\mathrm{c}}x_{\mathrm{T}}}{\partial \boldsymbol{p}} & \frac{\partial\, {}^{\mathrm{c}}y_{\mathrm{T}}}{\partial \boldsymbol{p}} \end{bmatrix}^{\mathrm{T}},
\end{aligned}
\tag{S14}
$$

$$
\begin{aligned}
\frac{\partial\, {}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{Q}}}}{\partial \boldsymbol{p}} &= \begin{bmatrix} \frac{\partial\, {}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{Q}}}}{\partial\, {}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{P}}}} & \frac{\partial\, {}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{Q}}}}{\partial\, {}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{S}}}} & \frac{\partial\, {}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{Q}}}}{\partial\, {}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{Q}}} & \frac{\partial\, {}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{Q}}}}{\partial\, {}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{T}}} \end{bmatrix} \begin{bmatrix} \frac{\partial\, {}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{P}}}}{\partial \boldsymbol{p}} & \frac{\partial\, {}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{S}}}}{\partial \boldsymbol{p}} & \frac{\partial\, {}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{Q}}}{\partial \boldsymbol{p}} & \frac{\partial\, {}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{T}}}{\partial \boldsymbol{p}} \end{bmatrix}^{\mathrm{T}}, \\
\frac{\partial\, {}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{T}}}}{\partial \boldsymbol{p}} &= \begin{bmatrix} \frac{\partial\, {}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{T}}}}{\partial\, {}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{P}}}} & \frac{\partial\, {}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{T}}}}{\partial\, {}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{S}}}} & \frac{\partial\, {}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{T}}}}{\partial\, {}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{Q}}} & \frac{\partial\, {}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{T}}}}{\partial\, {}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{T}}} \end{bmatrix} \begin{bmatrix} \frac{\partial\, {}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{P}}}}{\partial \boldsymbol{p}} & \frac{\partial\, {}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{S}}}}{\partial \boldsymbol{p}} & \frac{\partial\, {}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{Q}}}{\partial \boldsymbol{p}} & \frac{\partial\, {}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{T}}}{\partial \boldsymbol{p}} \end{bmatrix}^{\mathrm{T}}.
\end{aligned}
\tag{S15}
$$

Here at the end of the obstacle-set method, we have obtained ${}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{P}}}(\boldsymbol{p})$, ${}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{S}}}(\boldsymbol{p})$, ${}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{Q}}}(\boldsymbol{p})$, and ${}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{T}}}(\boldsymbol{p})$ as well as their gradients. Before computing moment arm with (13), we need to utilize (S1) to transform them from their respective joint frames to the world frame.

Finally, (4) may now be extended into

$$
\frac{\partial \boldsymbol{r}}{\partial \boldsymbol{p}} = \begin{bmatrix} \frac{\partial \boldsymbol{r}}{\partial\, {}^{\mathrm{w}}\boldsymbol{u}_{\tilde{\mathrm{P}}}} & \frac{\partial \boldsymbol{r}}{\partial\, {}^{\mathrm{w}}\boldsymbol{u}_{\tilde{\mathrm{S}}}} & \frac{\partial \boldsymbol{r}}{\partial\, {}^{\mathrm{w}}\boldsymbol{u}_{\tilde{\mathrm{Q}}}} & \frac{\partial \boldsymbol{r}}{\partial\, {}^{\mathrm{w}}\boldsymbol{u}_{\tilde{\mathrm{T}}}} \end{bmatrix} \begin{bmatrix} \frac{\partial\, {}^{\mathrm{w}}\boldsymbol{u}_{\mathrm{P}}}{\partial \boldsymbol{p}} & \frac{\partial\, {}^{\mathrm{w}}\boldsymbol{u}_{\mathrm{S}}}{\partial \boldsymbol{p}} & \frac{\partial\, {}^{\mathrm{w}}\boldsymbol{u}_{\tilde{\mathrm{Q}}}}{\partial \boldsymbol{p}} & \frac{\partial\, {}^{\mathrm{w}}\boldsymbol{u}_{\tilde{\mathrm{T}}}}{\partial \boldsymbol{p}} \end{bmatrix}^{\mathrm{T}},
\tag{S16}
$$

where the first term on the right side is easily derived from (13), while the second term lies in the chain of (S7), (S12), and (S12)–S15. The derivation of this compound gradient is verified as described in Subsection II-C.

D. Nomenclature

S TABLE I
NOMENCLATURE FOR 12 DOFS IN THE HUMAN SHOULDER AND ARM [40].

Joint	DoF (Moment Arm)
Sternoclavicular 1	Protraction/retraction
Sternoclavicular 2	Elevation/depression
Acromioclavicular 1	Lateral abduction/adduction
Acromioclavicular 2	Internal/external rotation
Acromioclavicular 3	Posterior abduction/adduction
Glenohumeral 1	Anteversion/retroversion
Glenohumeral 2	Abduction/adduction
Glenohumeral 3	Interior/exterior rotation
Humeroulnar	Flexion/extension
Radioulnar	Pronation/supnation
Radiocarpal 1	Palmarflexion/dorsiflexion
Radiocarpal 2	Radial/ulnar abduction

S TABLE II

NOMENCLATURE AND PARAMETER STRUCTURE FOR 42 MUSCLE PATHS IN THE HUMAN SHOULDER AND ARM.

Acronym	**Muscle Path**	**Parameter Structure**[1]	
		Original	Modified
SBCL	subclavius	OI (6)	OCI (12)
SRAs	serratus anterior (superior)	OCI (12)	OCI (12)
SRAm	serratus anterior (middle)	OCI (12)	OCI (12)
SRAi	serratus anterior (inferior)	OCI (12)	OCI (12)
TRPc	trapezius (C1–C6)	OCI (12)	OCI (12)
TRPc7	trapezius (C7)	OCI (12)	OCI (12)
TRPt1	trapezius (T1)	OCI (12)	OCI (12)
TRPt	trapezius (T2–T7)	OCI (12)	OCI (12)
LVS	levator scapulae	OI (6)	OCI (12)
RMN	rhomboid minor	OCI (12)	OCI (12)
RMJt2	rhomboid major (T1–T2)	OCI (12)	OCI (12)
RMJt3	rhomboid major (T3–T4)	OCI (12)	OCI (12)
PMN	pectoralis minor	OI (6)	OCI (12)
PMJc	pectoralis major (clavicularis)	OCI (12)	OCI (12)
PMJs	pectoralis major (sternocostalis)	OCI (12)	OCI (12)
PMJr	pectoralis major (abdominalis)	OCI (12)	OCI (12)
LTDt	latissimus dorsi (thoracic)	OCCI (18)	OCCI (18)
LTDl	latissimus dorsi (lumbar)	OCCI (18)	OCCI (18)
LTDi	latissimus dorsi (iliac)	OCCI (18)	OCCI (18)
DLTc	deltoid (clavicular)	OCVI (15)	OCCI (18)
DLTa	deltoid (acrominal)	OVCVI (18)	OCCI (18)
DLTs	deltoid (scapular)	OVCVI (18)	OCCI (18)
SUPR	supraspinatus	OVVI (12)	OCCI (18)
INFR	infraspinatus	OVCI (15)	OCI (12)
SBSC	subscapularis	OVCI (15)	OCI (12)
TMN	teres minor	OVCI (15)	OCI (12)
TMJ	teres major	OCI (12)	OCI (12)
CRCB	coracobrachialis	OCVI (15)	OCI (12)
TRClg	triceps brachii (long head)	OCCI (18)	OCCI (18)
TRCm	triceps brachii (medial)	OVCI (15)	OCI (12)
TRClt	triceps brachii (lateral)	OVCVI (18)	OCI (12)
BICsh	biceps (short head)	OCCI (18)	OCCI (18)
BIClg	biceps (long head)	OCVCI (21)	OCCI (18)
BRA	brachialis	OVCI (15)	OCI (12)
BRD	brachioradialis	OVCCVI (24)	OCCI (18)
SUP	supinator	OCVI (15)	OCI (12)
PRNT	pronator teres	OVCI (15)	OCI (12)
FCR	flexor carpi radialis	OVCVI (18)	OCI (12)
FCU	flexor carpi ulnaris	OVCI (15)	OCI (12)
ECRL	extensor carpi radialis longus	OVCVVCVI (30)	OCCI (18)
ECRB	extensor carpi radialis brevis	OCVCVI (24)	OCCI (18)
ECU	extensor carpi ulnaris	OVVVCCI (24)	OCI (12)

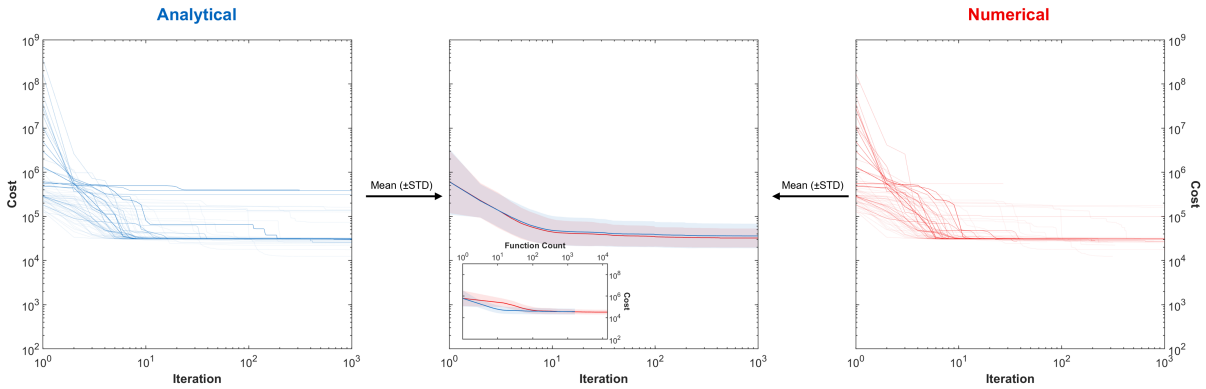
[1]Denoted by a combination of the origin (O), via (V), and insertion (I) points, as well as cylinders (C) in the specified order. The number of parameters is shown in the bracket following each structure. The original structure is defined by [18].

E. Supplementary Figures (Analytical vs. Numerical)

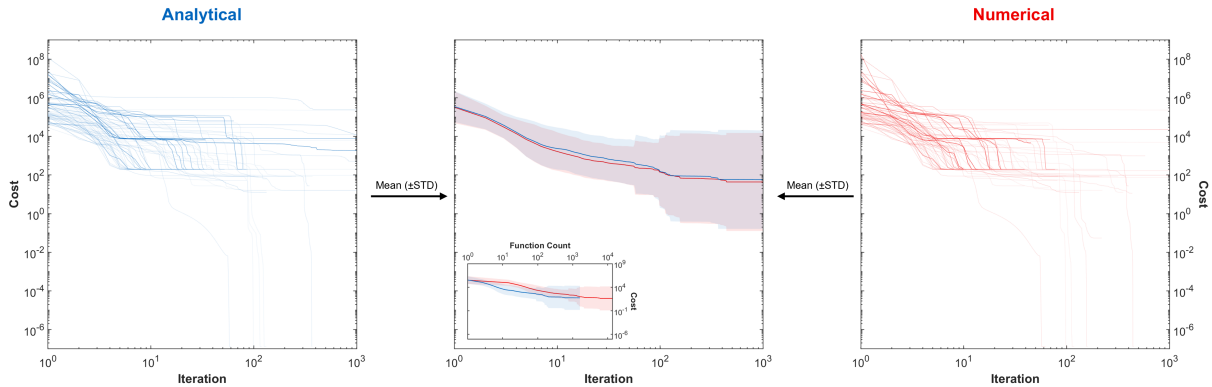
The following figures are supplements to Figs. 2 and 3, showing the cost–iteration curves for the optimization using either the algorithm of

1) `levenberg-marquardt` (70 initial points), or
2) `trust-region-reflective` (84 initial points).

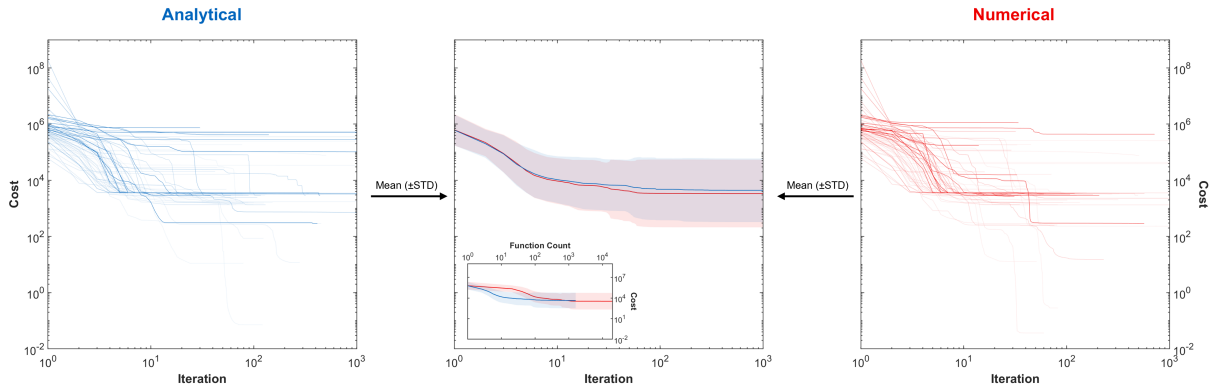
The results of each initial point from the analytical (blue) and numerical (red) gradients are plotted respectively on the left and right. The mean and STD of the costs are plotted in the middle, with function count–dependent curves in the corner.



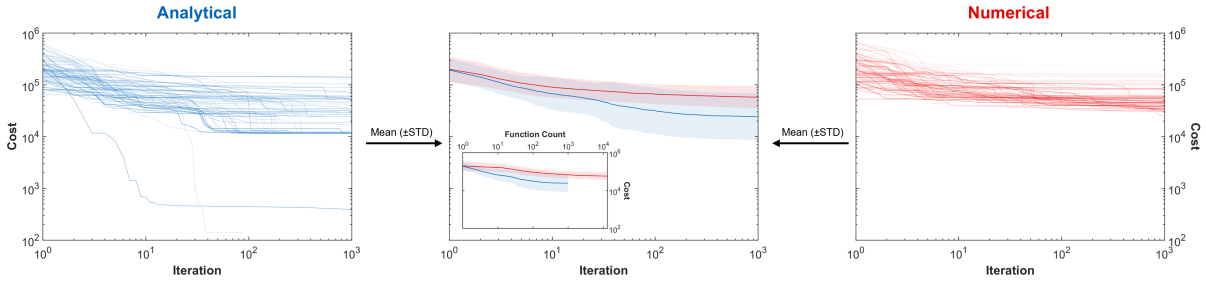
S Fig. 3. `levenberg-marquardt`, serratus anterior (superior part).



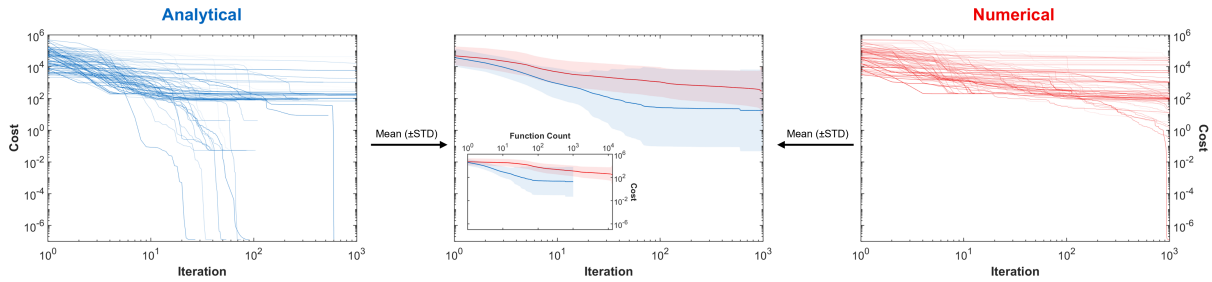
S Fig. 4. `levenberg-marquardt`, serratus anterior (middle part).



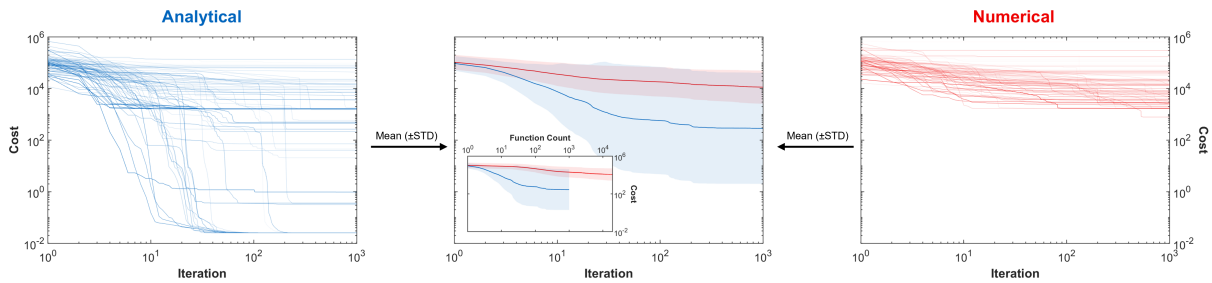
S Fig. 5. `levenberg-marquardt`, latissimus dorsi (thoracic part).



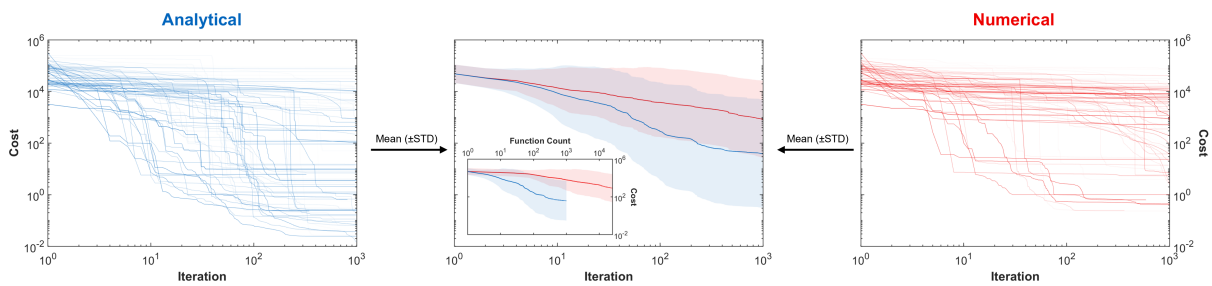
S Fig. 6. `trust-region-reflective`, serratus anterior (superior part).



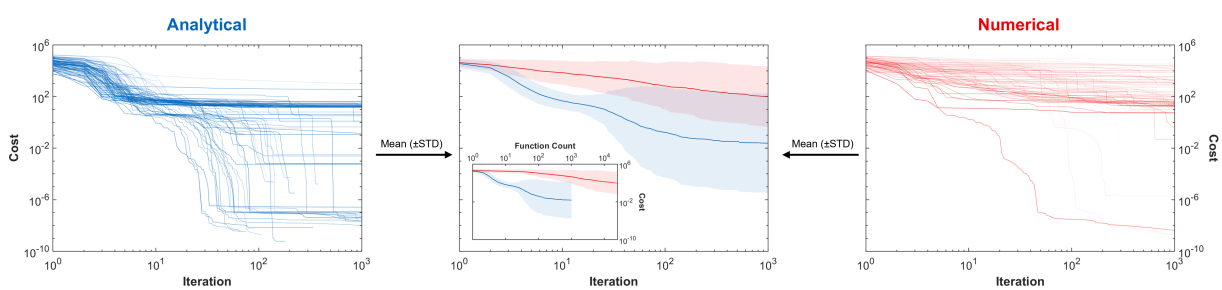
S Fig. 7. `trust-region-reflective`, serratus anterior (middle part).



S Fig. 8. `trust-region-reflective`, latissimus dorsi (thoracic part).



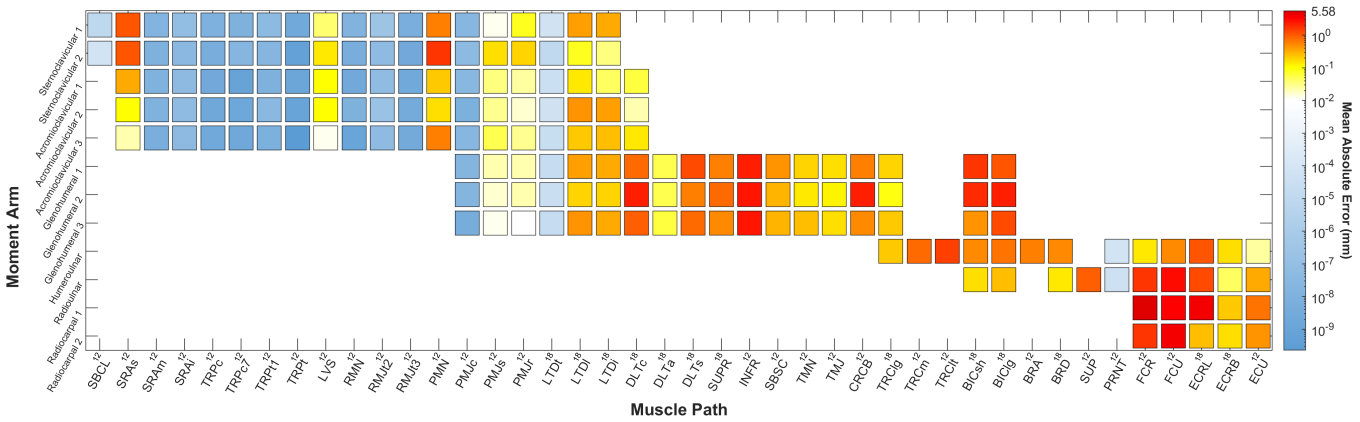
S Fig. 9. `trust-region-reflective`, extensor carpi radialis brevis.



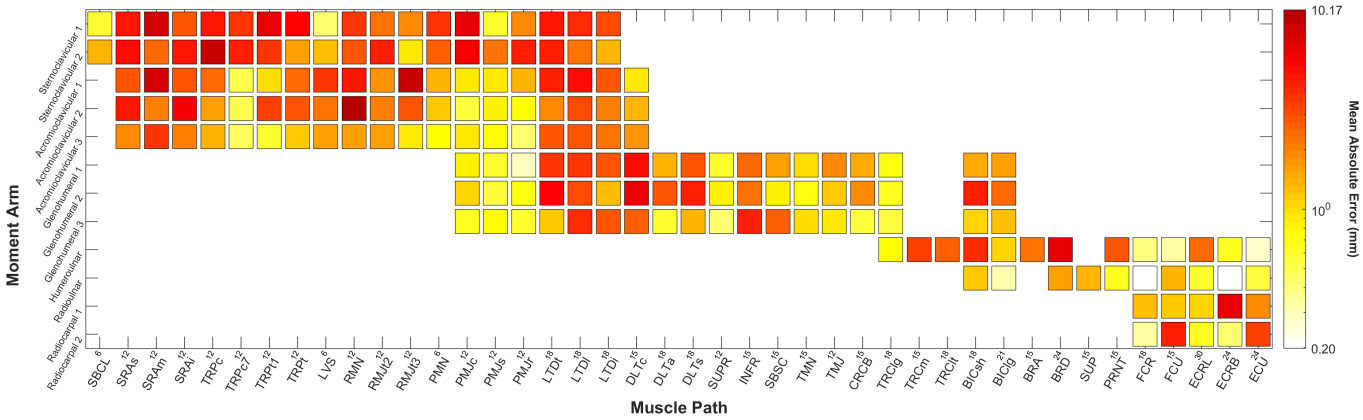
S Fig. 10. `trust-region-reflective`, extensor carpi ulnaris.

F. Supplementary Figures (Validation Error)

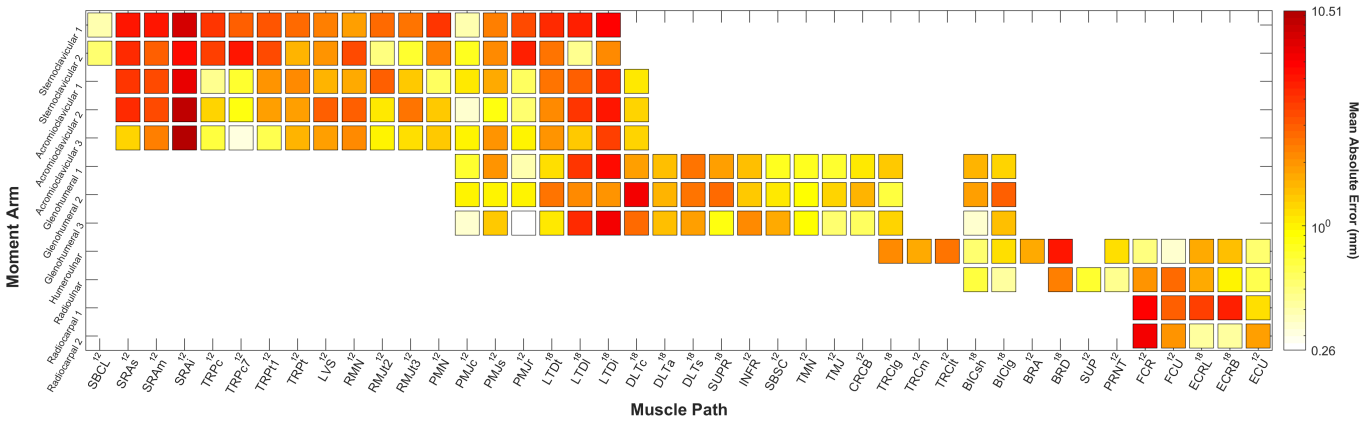
The following figures are supplements to Fig. 5, showing the validation errors in 182 effective moment arms of 42 muscle paths. The magnitude of the error is indicated by the intensity of the color. The number in the horizontal axis corresponds to the parameter number for each muscle path. See Appendix D for nomenclatures in the axes.



S Fig. 11. Modified parameter structure, noise-free calibration data.



S Fig. 12. Original parameter structure, noisy calibration data.



S Fig. 13. Modified parameter structure, noisy calibration data.